JULY 1990 £2.75

Commodore DISK·USER

GRAPHICS FACTORY

Enter a new dimension in graphics designing

Adventure Helpline • Arcadia Life in a Coin-Op • DIY Centronics Interface

ISSN 0953-0614

07

9 770953 061007

THE ULTIMATE CARTRIDGE COMES OF AGE!

ONLY
£34.99
POST FREE

THE ACTION REPLAY Mk VI WILL LOAD A 200 BLOCK PROGRAM IN UNDER 6 SECONDS

NOW ACTION REPLAY MK VI IS HERE!

THE MOST POWERFUL, FRIENDLY AND FEATURE PACKED UTILITY CARTRIDGE EVER CONCEIVED!

- **TURBO LOADER** - Load 202 block program in under 6 seconds - world's fastest disk serial loader. On-board RAM and ROM achieves high loading speeds. Works with 1541/1571/Oceanic/1581.
- **INFINITE LIVES GENERATOR** - Automatic infinite lives!! Very easy to use, works with many programs. No user knowledge required.
- **PROF MACHINE CODE MONITOR** - Full 64K Freezer Monitor - examine ALL memory, including stack, I/O area and registers in their frozen state. Ideal for de-bugging or just for fun!
- **SPRITE CONTROL** - Freeze the action and view the sprites - watch the animation - customise your games - kill sprite collisions.
- **FREEZER FACILITY** - Now you can make your old slow loading programs load faster. Simply freeze the action and save to tape or disk to reload, independently, at superfast speed - no more waiting for programs to load.
- **DISK COPY** - Easy to use disk/file copier. Much faster than conventional methods. Ideal for backing up data disks.
- **TAPE TURBO** - This feature will add Turbo Reload to the programs that you save to tape - no user knowledge required.
- **FAST FORMAT** - Format an entire disk in about 10 seconds - no more messing about.
- **PRINTER DUMP** - Print out your frozen screen to printer - MPS 801, 803, Epson, Star, etc. - very versatile.
- **CENTRONICS INTERFACE** - For parallel printers, Star, Epson, etc. Print out listings with graphic characters etc. (Cable required for parallel port £12.99).
- **SCREEN EDITOR** - Now you can edit the entire frozen screen with this text editor - change names on high scores, etc. Great fun!!
- **EXTENDED TOOLKIT** - Many single stroke commands for Load, Save, Dir, etc. Plus range of extra commands, i.e. Auto Number, Old, Delete, Merge, Append, Linesave, etc.

GRAPHICS SUPPORT UTILITIES DISK

SLIDE SHOW - View your favourite screens in a slide show type display

BLOW UP - Unique utility allows you to take any part of a picture & "blow it up" to full screen size

SPRITE EDITOR - A complete sprite editor helps you to create or edit sprites

MESSAGE MAKER - Any screen captured with Action Replay or created with a graphics package can be turned into a scrolling screen message with music

ONLY £9.99

THE REVIEWERS SAID...

"I'm stunned, amazed and totally impressed. This is easily the best value for money cartridge. *THE CARTRIDGE KING!*"

COMMODORE DISK USER

HOW TO GET YOUR ACTION REPLAY MK VI

TELEPHONE (24 Hrs) - 0782 744707 - CREDIT CARD ORDERS

WE WILL DESPATCH YOUR ORDER QUICKLY & EFFICIENTLY TO ENABLE YOU TO START RECEIVING THE BENEFITS OF YOUR PURCHASE WITHIN DAYS, NOT WEEKS.

ORDERS NORMALLY DESPATCHED WITHIN 48 Hrs. ALL CHEQUES/POSTAL ORDERS MADE PAYABLE TO...

Access VISA

DATEL ELECTRONICS LTD.,

GOVAN ROAD, FENTON INDUSTRIAL ESTATE, FENTON, STOKE-ON-TRENT, ST4 2RS, ENGLAND

TECHNICAL/CUSTOMER SERVICE 0782 744324

CONTENTS

IN THE MAGAZINE

ON THE DISK

Commodore Disk User
Volume 3 Number 9
July 1990

AN Argus SPECIALIST PUBLICATION

Editor: PAUL EVES
Group Editor: STUART COOKE
Assistant Editor: HILARY CURTIS
Photography: MANNY CEFAI
Adventure Correspondent: GORDON HAMLETT
Advertisement Manager: PAUL KAVANAGH
Display Sales Exec: MARIA WADE
Classified Sales Exec: TONY FLANAGAN
Designer: MARK NEWTON
Origination: EBONY TYPESETTING
Distribution: S. M. DISTRIBUTION
Printed By: CHASE WEB LTD, ST IVES PLC

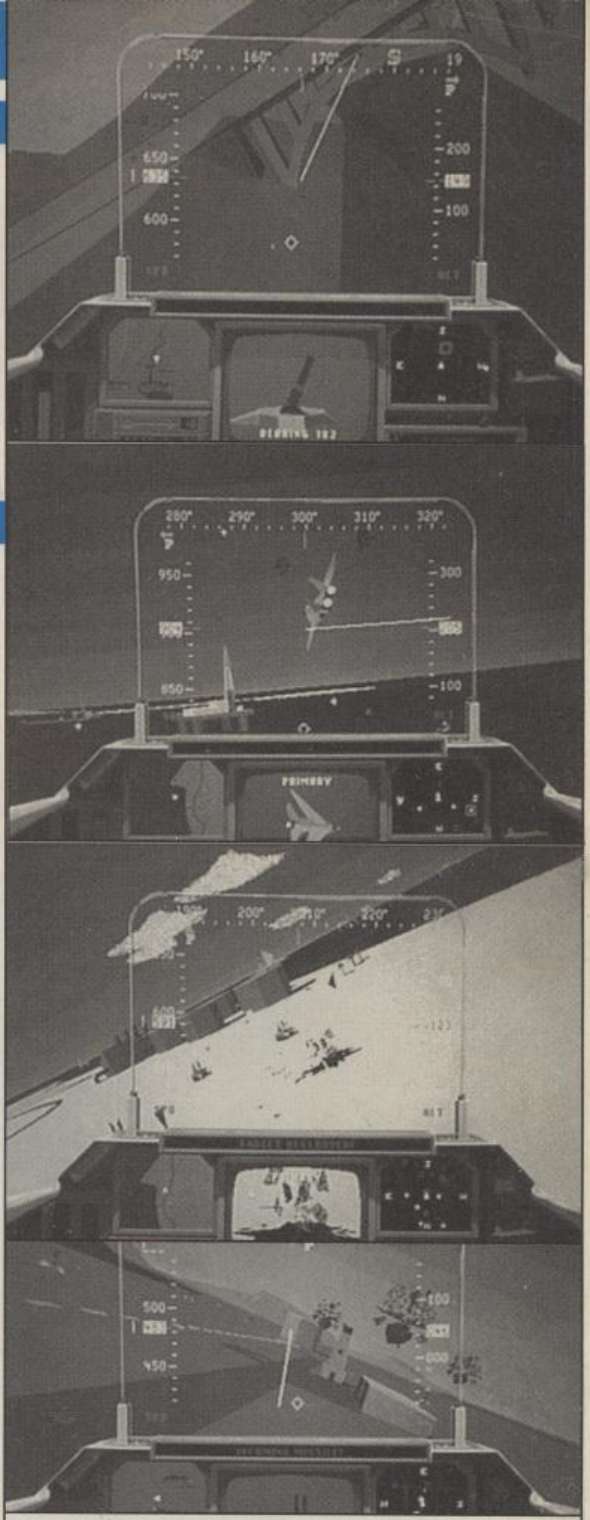
SUBSCRIPTION RATES

Here are the rates for subscriptions to CDU with effect from November 1989

UK	**£33.00**
Europe	**£39.00**
Middle East	**£39.30**
Far East	**£41.60**
Rest of the World £39.70 or	**$69.00**

Airmail rates on request

Commodore Disk User is a monthly magazine published on the 3rd Friday of every month. **Argus Specialist Publications, Argus House, Boundary Way, Hemel Hempstead, HP2 7ST. Telephone: (0442) 66551 Fax: (0442) 66998.**



A preview of what is in this month's issue.

Important announcement

Dear Readers,
Unfortunately this issue of CDU will mark the last time the magazine will be published. As from the JULY cover dated issue, CDU will no longer be in existence.

I regret this decision which has come from the upper echelons of the building, but, unfortunately, there seems to be nothing I can do.

The overall 8 bit market is suffering from the onslaught of the 16 bit machines and as a result sales are falling to drastic levels. obviously any company in business is in business to make profits, if profits no longer exist then something has to be done.

I must say that it has all been rather a shock to me as I was under the impression that CDU was doing very nicely, and if your letters are anything to go by, then it is, or certainly has been.

I don't want to labour the point, so I will just say thank you for all your support and interest in the past. I wish everyone of you the very best of luck.

If any one would like to write to me for any reason, be it for suggestions, programming tips etc then I can be reached at:
26, Ridgeway
Berkhamsted
Herts
HP4 3LD

Once again, I thank you for all your support.

Disk Instructions

Although we do everything possible to ensure that CDU is compatible with all C64 and C128 computers, one point we must make clear is this. The use of 'Fast Loaders', 'Cartridges' or alternative operating systems, such as Dolphin DOS, may not guarantee that your disk will function properly. If you experience problems and you have one of the above, then we suggest you disable them and use the computer under normal, standard conditions. Getting the programs up and running should not present you with any difficulties, simply put your disk in the drive and enter the command.

Back issues of CDU are available at £3.25 per issue, which includes postage and packing via:

Select Subscriptions Ltd
5, River Park Estate
Berkhamsted
Herts
HP4 1HL
Tel: 0442-876661

VOL 2 No. 4 MAY/JUN 89
BASE ED – Get organised with this C64 database.
DBASE 128 - 40 or 80 column storage for C128 owners.
6510+ – The ultimate in C64 assembly programs.
SID SEQUENCER – Make Commodore music with ease.
LIBERTE – Escape the POW camp in this 1940's style adventure.
FX KIT – Bangs, Pows and Zaps made easy.

VOL 2 NO.5 JUL/AUG 89
FONT FACTORY – Create your own chars.
HI-RES DEMO KIT – Add music to your favourite picture.
ANIMATOR – Get those sprites moving.
BORDER MESSAGE SCROLL – Say what you want along the bottom of the screen.
TYPIT 128 – Create professional text layouts on your C128.
SCREEN COPIES UTITLITY – Download your favourite screens.
VIDI-BASIC – Graphic based extension to Basic.
64 NEWS DESK – Become a C64 reporter.

VOL 2 No.6 SEP/OCT 89
MICKMON – An extensive M/C monitor.
SCRAPBOOK – Collectors and hobbyists database.
CELLRATOR – Enter the caves if you dare.
RAINBOW CHASER – Rainbows means points in the unusual game.
HIDDEN GRAPHICS – Utilise those graphic secrets.
FORTRESS – Save the world. Yet again!!
DISK HUNTER – Keep tabs on your disk library.
SUPERFILE – One more for the record keepers.

VOL 3 N0.1 NOVEMBER 89
BASIC EXTENSION – Windows and Icons the easy way
B-RAID – Vertical scrolling shoot 'em up
DISKONOMISER – Prudent disk block saving
HELP – Design your own information help screens
ORSITAL – An arcade style game with a difference
PROGRAM COMPARE – Modifying Basic progs has never been easier
RASTER ROUTINES – A few colourful demos
SPRITE EDITOR 1 – A no nonesence basic sprite editor
WABBIT – Help the rabbit collect his carrots

VOL 3 No.3 JANUARY 90
4 IN A ROW – Connect a row of counters
FROGS IN SPACE – Leap to safety across the space lanes
BLACKJACK – Don't lose your shirt
LORD OF DARKNESS – Defeat the evil lord true adventure style
MARGO – Fly around and collect the jewels
JETRACE 2000 – Have you got what it takes to be best
ULTIMATE FONT EDITOR – Create your own screens and layouts
SELECTIVE COLOUR RESTORE – Design your own start up colours
6510+ UNASSEMBLER – Transform M/C into Source, with labels
TRIVIA CHALLENGE – The first of 3 files for this superb game

VOL 3 No.4 FEBRUARY 90
COLOUR PICTURE PRINT – Download your favourite colour screens
BASE-ED 2 – An update to our popular database system
1ST MILLION – Play the market in this strategy game
FM-DOS – Enhance your drives oper-

LOAD "MENU", 8, 1

Once the disk menu has loaded you will be able to start any of the programs simply by selecting the desired one from the list. It is possible for some programs to alter the computer's memory so that you will not be able to LOAD programs from the menu correctly until you reset the machine. We therefore suggest that you turn your computer off and then on again, before loading each program.

How to copy CDU files

You are welcome to make as many of your own copies of CDU programs as you want, as long as you do not pass them on to other people, or worse, sell them for profit. For people who want to make legitimate copies, we have provided a simple machine code file copier. To use it, simply select the item FILE COPIER from the main menu. Instructions are presented on screen.

Disk Failure

If for any reason the disk with your copy of CDU will not work on your system then please carefully re-read the operating instructions in the magazine. If you still experience problems then:

1) If you are a subscriber, return it to:
Select Subscriptions Ltd
5, River Park Estate
Berkhamstead
HERTS
HP4 1HL
Tel: 0442-876661

2) If you bought it from a newsagents, then return it to:
CDU Replacements
Protoscan Europe PLC
Burrell Road
St. Ives
Cambs
P17 4LE
Tel: 0480-495520

(Within eight weeks of publication date disks are replaced free).

After eight weeks a replacement disk can be supplied from Protoscan for a service charge of £1.00. Return the faulty disk with a cheque or postal order made out to Protoscan and clearly state the issue of CDU that you require. No documentation will be provided.

Please use appropriate packaging, cardboard stiffener at least, when returning disk. Do not send back your magazine-only the disk please.

NOTE: Do not send your disks back to the above if its a program that does not appear to work. Only if the DISK is faulty. Program faults should be sent to the editorial office marked FAO bug-finders. Thank you.

Back Issues

ating system
GEOS FONTS – A further 4 fonts for Geos users
HASHING IT – Reletive filing made easy
MULTI-SPRITE – Make full use of up to 24 sprites
DIRECTORIES EXPLAINED – Find your way through the directory jungle
TRIVIA CHALLENGE – The second part of this popular game

VOL 3 No.5 MARCH 90
PLAGUE – Become your planets Guardian and Defender
SURROUND – Reversi on the C64
GEOS FONTS – The last of 12 new fonts
SCREEN SLIDE – Create your own slideshows
JOYSTICK TESTER – Put your stick(s) through the mill
COLOUR MATCHER – Mastermind for the younger ones
SCREEN MANIPULATOR – Full use of the screen now obtainable
TRIVIA CHALLENGE – The last of the 3 files for the game
VIDEO RECORDER PLANNER – Keep tab on your recordings

VOL 3 No.6 APRIL 90
BAR PROMPTS – M/C alternative input routine
HI-LITE BARS – as Bar Prompts but in Basic
TEXAS DEMO – Examples of using Basic for demos
CHARS TO SPRITES – Convert UDG's to sprites
FONT FACTORY – Compliments Chars to Sprites
3D-TEXT MACHINE – Impressive 3D text screens made easy
SCREEN ENHANCER – Makes full use of the screen easy to do
SPREADSHEET 64 – An excellent, easy to use spreadsheet
MINI-AID – 3 short utilities to aid the Basic programmer
C128 COLLECTION – 3 useful C128 programs

VOL 3 No.7 MAY 90
NUDGE – FLD explained and made easy
WINDOW WIPER – An alternative screen clear system
CHARACTER EXTRACTOR – Borrow those nice character sets
MAZE GENERATOR – Create your own fun
HIRES ANIMATOR – This difficult subject made easier
SPRITE DRIVER – Design your own platform games without fuss
ROTATRON – Demonstration of sprites and sounds
TEXT COMPRESSION – How to squeeze a gallon into a pint
SCREENS – make up your own help screens and keep them in memory
INTERRUPT POINTERS – Geos style windows and pointers for you

VOL 3 No. 8 JUNE 90
ALEATORY MUSIC – An alternative music system
SPRITE BASIC – Efficient sprite handling through basic
SPRITE GENERATOR – Another sprite editor for the library
MUNCHER – Pacman returns with a vengeance
ASTRODUS – Escape the spaceship Astrodus in this adventure
1581 DIRECT ACCESS – Find your way round the 1581 disk drive
PERSONAL ORGANISER – Design your own organiser pages
128 CONVERTOR/MATSH AID – 2 utilities for C128 users

All orders should be sent to: Select Subscriptions Ltd, 5, River Park Estate, Billett Lane, Berkhamsted, Herts, HP4 1HL. Please allow 28 days for delivery.

HARWOODS

YOUR FIRST CHOICE FOR AMIGA

POWE

Port

NEW! AMIGA POWERPLAY PACKS

At Gordon Harwoods we've yet again improved our Great Value Amiga Offers with the launch of our LATEST & GREATEST EVER POWERPLAY PACKS; there's EVEN MORE SOFTWARE *Plus* YOU NOW HAVE A CHOICE!

ALL OUR PACKS CONTAIN AMIGAS WITH THE FOLLOWING STANDARD FEATURES:-

- 512K RAM
- 1Mb Disk Drive
- 4096 Colours
- Multi-Tasking
- Built-in Speech Synthesis
- Mouse
- 3 Operation Manuals
- Workbench 1.3 System Disks
- Kickstart 1.3 Built-in
- All Connecting Cables

ALL OUR PRICES INCLUDE VAT AND STANDARD DELIVERY SERVICE

Now you can have 11 NEW SIZZLING SOFTWARE TITLES, EXCLUSIVE TO GORDON HARWOOD COMPUTERS, PLUS A CHOICE OF 3 MORE (Either Group A or Group B) below, when you buy Packs 1,2,3 & 5

GROUP A
- Batman the Movie
- New Zealand Story
- F18 Interceptor

GROUP B
- Escape from the Planet of the Robot Monster
- Rainbow Island
- F29 Retaliator

WHEN ORDERING...
Simply quote which 'Pack' you require along with your choice of Group 'A' or 'B' Software (eg. Pack 1 with Group B s/ware)

NEW! NEW! NEW! NEW! NEW! NEW!

Amiga POWERPLAY PACKS

NEW A & B Amiga PACK 1

THE COMPLETE PACK FOR THE GAMES ENTHUSIAST AVAILABLE RIGHT NOW!

Our ALL NEW Amiga Powerplay Pack 1 now includes some great up to the minute software and extras, just look at the savings you're going to make!!!

AMIGA A500 COMPUTER PLUS...

- Batman the Movie
- New Zealand Story
- F18 Interceptor

OR

- Escape from the Planet of the Robot Monster
- Rainbow Island
- F29 Retaliator

- Commando
- Running Man
- Nebulus
- Menace
- Chessmaster 2000
- Trivial Pursuit
- Tetris
- Phobia
- Voyager
- Archipelagos
- Pacmania
- Deluxe Paint II
- Microswitched Joystick
- Tailored Amiga Dust Cover
- Tutorial Disk
- TV Modulator (Packs 1 & 5 ONLY)
- Mouse Mat

SEE WHAT WE MEAN ABOUT A COMPLETE PACK!

You won't need to buy any more games for ages and you'll be able to start using your Amiga the moment it's unboxed!

£399

* OR SPREAD THE COST WITH OUR FINANCE FACILITIES

NEW A & B Amiga PACK 5

HARWOODS ALL NEW COMPLETE STUDENTS PACK

- Amiga A500
- Publishers Choice (DTP- Includes Kindwords V.2)
- Midi Master I/Face
- Dr. T's Midi Rec. Studio
- BBC Basic Emulator
- Maxiplan 500 S/sheet
- Superbase Personal - Database Program
- Amiga Logo
- Deluxe Paint II
- Microswitched Joystick
- 10-Blank 3.5" Disks
- Disk Wallet
- Mouse Mat
- TV Modulator

PLUS 14 GREAT GAMES TOO!!!
(See Pack 1 [→] for details of titles)

Superb Value from Harwoods...

Only... £549*

NEW A & B Amiga PACK 2

Containing the Super Powerplay Pack 1, AND a Commodore 1084S Stereo Colour Monitor PLUS a Free Tailored Monitor Dust Cover

See These Games for Only... £599*

NEW A & B Amiga PACK 3

NEW AMIGA AND COLOUR PRINTER PACK

Take our Powerplay Pack 2 and add Star's fantastic LC10 COLOUR PRINTER, to give you the ultimate colour home entertainment computer system!!!

Or, if you prefer an alternative printer from within our range, just deduct £209.95 and add the price of the printer you require. (Any printer can be chosen)

Superb Value... £799*

NEW! Amiga PACK 4

NEW POWERPRO PACK 4 CONSISTS OF...

Amiga A500, Star LC10 Colour Printer, Commodore 1084S Stereo Colour Monitor "Protext" Version 4.2 Word Processor "Superbase II" Database "Maxiplan 500" Spreadsheet Deluxe Paint II, Mouse Mat Ten 3.5" Blank Disks & Library Case Quality Tailored Dust Covers for Amiga, Monitor & Star Printer

If you prefer an alternative printer from our range, deduct £209.95 & add the price of the replacement.

£829*

HARWOODS THE NAME YOU CAN TRUST

Access VISA

F

FINANCE FACILITIES

Gordon Harwood Computers offer facilities to pay by our budget account scheme for most items. APR 34.5% (Variable) Credit sale terms are available to most adults of eighteen years or over, subject to status. Simply phone or write and we will send written details along with an application form. Applications are required in advance and are available to full time U.K. mainland residents only.

Written quotations available on request.

* **Credit sale terms with or without a deposit, can be tailored to suit your needs.**

ORDERING MADE EASY - COMPARE OUR SERVICE

ORDER BY PHONE: Phone our Order Hotline with your Mastercard, Access, Visa or Lombard Charge Card quoting number & expiry date.

ORDER BY POST: Make cheques, bankers-building society drafts or postal orders payable to GORDON HARWOOD COMPUTERS. (Personal or business cheques take 7days to clear from day of receipt before your order is despatched)
Please send Name, Address, and most importantly if possible, a Daytime Telephone Number along with your order requirements.

TO PAY BY GIRO TRANSFER: at your own bank, simply phone us for details (Transfers normally take 3-5 working days).

FREE POSTAL DELIVERY: Goods in UK Mainland (5-7 day delivery). **OR COURIER SERVICE** ...Add £5 per major item for next working day delivery, UK Mainland most regions.
(Goods normally despatched on day of ordering or payment clearance)

EXPORT ORDERS: Most items are available at TAX FREE PRICES to non UK residents when visiting us or for export shipment. Please contact us for ordering information.

REMEMBER AFTER YOU'VE BOUGHT FROM HARWOODS, WE'LL STILL BE HERE...

TECHNICAL SUPPORT: You will be given our Exclusive Technical Support Phone Number to call should you require any help or advice on any aspect of the system you have purchased.

12 MONTH WARRANTY: Items proving faulty within 30 days of purchase will be replaced with NEW UNITS. For the whole guarantee period, all warranty service will be completely FREE OF CHARGE and a FAST turnaround is GUARANTEED by our OWN ENGINEERS!

COLLECTION FACILITY: Any computer, monitor or printer requiring warranty service can be collected from your home FREE OF CHARGE during the guarantee period (UK Mainland only)

FULL TESTING PROCEDURE: All computers are tested prior to despatch, and all items are supplied with mains plug and leads as required - just connect up and use straight away.

VERPLAY
ortfolio

MONITORS

PHILIPS CM 8833 STEREO

Full 14" Stereo High Resolution Colour Monitor

- SCART Euro-connector fitted as standard
- RGB/AI, TTL, Comp. Video & stereo audio inputs
- Can also be used as a TV with tuner or VCR
- Features fold down tilt-stand
- Twin Speakers
- FREE lead for computer of your choice
- ONLY FROM HARWOODS...12 Month replacement warranty for major faults

All this and a Free Tailored Dust Cover...

£249

COMMODORE 1084 S

Full 14" Stereo High Resolution Colour Monitor

- RGB/AI, TTL, Composite Video & Audio Inputs
- Can also be used as a TV with a tuner or VCR
- Features fold down tilt stand
- Twin Speakers
- Supplied with cables for A500, CGA PC, C16-64-128

GREAT NEW LOW PRICE!!!

With Free Tailored Dust Cover...

£239

GRAPHICS HARDWARE

DIGIVIEW GOLD 4.0

NEW VERSION 4.0 ENHANCED SOFTWARE

Digitises static images in Black and White or FULL COLOUR (using filters provided) and all resolutions supported (memory permitting). Creates IFF and HAM files. Uses B&W (or colour with B&W mode) video cameras.

NEW LOWER PRICE... £119.95

HITACHI VIDEO CAMERA

Mono, 650 scan lines. For photographic quality digitising with Digiview! *New Lower Price!*

£234.95

ILLUMINATED COPY STAND

Four-Light adjustable 'shake free' stand for your video camera. Makes digitising so much more accurate and simple to perform. Professionals choice at...

£99.95

COLOUR PIC

REAL TIME FRAME GRABBER

Instant colour frame grabber from moving video. Capture superb digitised video images in a range of resolutions from standard composite video source such as domestic video recorder or video camera in 50th of a second.

Super Price.....£439.95

RENDALE GENLOCKS

8802 SEMI-PROFESSIONAL £189.95

Complete with SOFTWARE CONTROL CABLE & BNC to PHONO CABLE!!!!

8806 PROFESSIONAL £749.00

ALL IN DIGITISING KIT

Consists of...DIGIVIEW GOLD 4.0 ILLUMINATED COPY STAND AND AN HITACHI VIDEO CAMERA WITH CABLE

Special Kit Price....£439.95

STORAGE DEVICES

HARD DRIVES

COMMODORE A590 20Mb HARD DISK DRIVE FOR AMIGA A500, WITH FREE 0.5 Mb MEMORY

Commodore's own hard drive for the A500

- Autoboot with Kickstart 1.3
- Sockets for up to 2Mb RAM expansion
- DMA access
- External SCSI port
- SUPPLIED WITH A FREE...0.5Mb MEMORY EXPANSION

BUY YOUR A590 FROM HARWOODS AND IT ALSO COMES COMPLETE WITH SOME FANTASTIC 1Mb SOFTWARE ABSOLUTELY FREE!

WORTH OVER £100!

FANTAVISION...Animation Package
COMIC SETTER...Cartoon Creation (Design your own comic book cartoons and print them too!)
SPRITZ PAINT...Paint Program

Super Low Price...

£399

A590 HARD DISK (As Above) WITH 2Mb MEMORY

- Expansion fitted and ready to use straight away

NEW LOW PRICE ONLY...£499.00!

40Mb VORTEX HARD DRIVE FOR BOTH AMIGA A500 AND AMIGA A1000

- For both A500 and A1000 as supplied
- Autoboots on ANY Amiga

GREAT VALUE AT ONLY...£499.00

FLOPPY DRIVES

EXTERNAL FLOPPY DISK DRIVES

The drives listed below have the following features...

- Enable/Disable switch
- Throughport
- LED Access light, super quiet
- Suitable for A500, A1000, A2000

CUMANA CAX 354 3.5" SECOND DRIVE

- 25mm Super slimline 3.5" drive

A REAL BARGAIN AT ONLY...£89.95

CUMANA CAX 1000S 5.25" SECOND DRIVE

- Amiga DOS and MS DOS compatible

SAVE MORE THAN EVER...£129.95

HARWOODS SUPERSLIM 3.5" SECOND DRIVE

- Extra storage and great value too!

'SUPERSLIM PRICE' ONLY...£64.95

ACCESSORIES

COMMODORE

A501 RAM PACK £99.95

GENUINE CBM ram pack with real time battery backed clock. This add-on DOES NOT invalidate Commodore's warranty.

A520 TV MODULATOR £19.95

QUALITY ACCESSORIES

NAKSHA Microswitched Mouse £24.95

FLOPPY DISKETTES

QUALITY CERTIFIED 3.5" BULK DISKS (supplied with labels)

Ten, with library case	£9.49
Ten, uncased	£8.49
Eighty, uncased	£49.95
Hundred, uncased	£59.95
Eighty with LOCKABLE storage case	£54.95

PHONE FOR LARGER QUANTITY DISCOUNTS!

PRINTERS

All printers in our range are Dot-Matrix and include the following features...

Standard Centronics parallel port for direct connection to Amiga, PC's, ST etc. and come with... FREE CONNECTOR CABLES!

Free Tailored Dust Cover With Every Star Printer Bought!

STAR LC10 - 120/25 cps

Our most popular Mono Dot-Matrix printer and at a super low price **£169.95**

STAR LC10 MkII - 150/30 cps

Brand new superfast MkII version of the ever popular LC10 printer.

- Multiple font options from front panel
- Excellent paper handling
- Simultaneous, continuous and single sheet stationery

£184.95

STAR LC10 COLOUR - 120/25 cps

Colour version of the LC10...

- Allows full colour dumps from Amiga
- Superb text quality
- Can use black LC10 ribbons
- Our most popular colour printer!

£209.95

STAR LC 24 10 - 180/60 cps

24 Pin version of the Star LC Series with

- Exceptional letter print quality

£249.95

STAR XB24 10 COLOUR - 240/80 cps

Very fast LQ 24 Pin Colour

- Includes on-site maintenance for a FULL 12 MONTHS!

£479.95

STAR POSTSCRIPT LASER - 8ppm C/W 35 FONTS

- 300 DPI, 2Mb. Memory upgradable to 5Mb.
- Emulations include: HP Series II, Epson EX800, IBM Proprinter and Diablo 630
- Includes on-site maintenancefor a FULL 12 MONTHS!
- Includes cable and courier delivery (3-4 days)

£2250.00

CITIZEN SWIFT 24 PIN COLOUR - 160/50 cps

The MOST competitive 24 Pin Colour Printer available Today!!! **£349.95**

CITIZEN 120D - 120/25 cps **2 YR WARRANTY**

Very reliable low cost printer, interchangeable interfaces available for Centronics RS 232 or Serial Type for CBM 64 etc. **£149.95**

NOTE! All our Star printers are genuine UK specification.

Characters Per Second speeds quoted are Draft/LQ at 10cpi

A FULL RANGE OF PRINTER RIBBONS ALWAYS AVAILABLE

Amiga SUNDRIES

DIEGO'S SPECIAL OFFERS...DIEGO'S SPECIAL OFFERS

MUSIC X SOFTWARE £109.95

Including Midi Interface....ABSOLUTELY FREE!!!

512K RAM UPGRADE £64.95

Including...'It Came from the Desert' and FREE... Mystery Major Amiga software title and Utility Software Disable Disk.

DEALIN' DIEGO YOUR AMIGA AMIGO ALWAYS SHOOTIN' OUT THE BEST DEALS!

FASTORDER HOTLINE-0773 836781

VISIT OUR SHOWROOM

Please call to see us where our full range of advertised products, AND MORE, is on sale. Come and see for yourself the amazing Amiga and a whole host of peripherals, software and accessories.

REMEMBER, WE ARE NOT JUST ANOTHER MAIL ORDER COMPANY.

All listed prices are what YOU PAY, and there are NO HIDDEN EXTRAS! VAT and postage are included and prices are correct at time of going to press (Pack details may vary from time to time). E & O E. Offers are subject to availability and are currently advertised prices.

GORDON HARWOOD HARWOOD HARWOOD Computers

GORDON HARWOOD COMPUTERS
DEPT,YAM1/B7, 69-71 HIGH STREET
ALFRETON, DERBYSHIRE, DE5 7DP
TEL:0773 836781 FAX:0773 831040

Adventure Helpline

Jason Finch gets down to more aid for all you adventurers stuck in the tale of KRON featured on the December disk

Last time I gave you enough clues, subtle or otherwise, to allow you to travel to the northern shore of Sark and to collect what you needed. Hopefully you will have completed that section and so we can move on to stage two of Kron, tackling the cave system and the guru – something I can admit to having been called in the past!!

The caves do indeed hold the key to your freedom but you are a long way from completing the adventure! Because caves are generally dark you will need some form of light. You have no matches but you should, if you followed last month's article, be able to do the old scout trick and obtain some fire. But little twigs soon burn out so there is something else that you should light, using the twigs as a sort of match to get the whole branch alight (whoops! Didn't mean to give away too many secrets!)

The system of caves is very complex with paths leading all over the place! You can explore them as much as you like but even withered branches don't burn forever. I will guide you to just the places you need to visit. Going to the east you will soon discover a silver mine where some unfortunate worker has perished. The picture holds some clue and although it isn't a particularly nice thing to do, search the skeleton and do whatever you think is necessary from there. Just remember to examine everything possible! The spade is, incidentally, of little significance.

When you have the nugget go west and twice south. You will find yourself by a stagnant pool – not very pleasant at all. Going east takes you to the Cave of Ice but don't get excited and rush there – your torch will go out and you haven't got the stone from the west of the lake yet! When you have got that, then you may go to the Cave of Ice!

Now you are near enough out of the caves and the torch goes out automatically because of the ice. So examine the things you have found. What was that lamp you found in the clam last month? Try rubbing it! You will be instructed to utter the words in gold. These can be found on the stone. Before the genie disappears (rub the lamp again if it does) say whatever it is you should and you will be told of further help that a man who lives alone upon a hill can provide. If you don't want help with this bit, now that I have given a little hint, then skip the next paragraph.

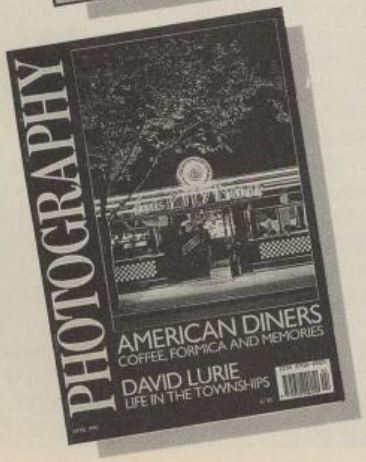
You must find this man! In the Cave of Ice you are able to jump to a higher passage. You see daylight above you. This time you should keep on climbing until you reach the top of the hill, finding yourself outside an old Boran hut. Don't do anything silly on the way like jumping into deep chasms, will you??! When you have sussed out how to get into the hut (what was the genie's message again?) You will have the Boran man at your disposal. The chances are that he went to the old Boran monastery in stage one and so may be able to understand what you found there but didn't have a clue as to what it meant. Offer it to him and see how it goes!

The next major problem you will face is frustration. Oh yes it is! Because when you have gone west to the Valley of the Dead, there seems no way on earth that you can cross the lake. Have you ever seen "The Eagle has Landed" or perhaps you like a bit of Galway. There you go – two hints telling you what you should do. I am just too kind! It is sufficient to say you need to be carried over.

But seeing as how you will possibly need a little time to figure it out I shall leave you to it and see you next month when I will help you through the third and final stage of this excellent adventure. And after that, who knows!

Subscribe now...

here's 4 good reasons why!



VIDEO TODAY

Published Monthly - SUBSCRIPTION PRICE **£19.20***

Britain's top selling video software magazine, VIDEO TODAY provides a highly successful blend of reviews, interviews with the stars, news and gossip on the video scene.
Full colour photographs and lively editorial make this magazine a must for all video owners.

WHICH VIDEO?

Published Monthly - SUBSCRIPTION PRICE **£18.00***

WHICH VIDEO? is aimed at all video users, especially those planning to upgrade or purchase equipment.
Each month WHICH VIDEO? publishes at least four-in-depth tests - these are supplemented by comprehensive consumer tests and features highlighting the latest developments and ideas in video technology. WHICH VIDEO? also features pre-recorede film reviews, readers technical queries, plus in-depth reviews and features on the world of satellite TV, as well as regular popular reader competitions.

FILM MONTHLY

Published Monthly - SUBSCRIPTION PRICE **£16.80***

FILM MONTHLY is a bright, lively and colourful screen magazine aimed exclusively at readers who love mainstream movie entertainment.
An informative monthly containing interviews with the leading celebrities, special behind-the-scenes reports on the latest films, news and gossip.
Also included is a detailed 16-page Review Section with reviews of topical big screen and video releases; books on showbiz; and film soundtrack albums.
With its editorial appeal to all ages backed up with excellent pictorial coverage, FILM MONTHLY puts readers firmly in the picture on what is happening in the world of movie and video entertainment.

PHOTOGRAPHY

Published Monthly - SUBSCRIPTION PRICE **£23.40***

Featuring world class photography at its best, PHOTOGRAPHY is internationally acclaimed for its stimulating content and quality presentation of mono and colour portfolios with indepth interviews with the world's leading photographers.
With its award winning design, PHOTOGRAPHY covers the full arena of art and is enjoyable as well as essential reading for photographers of all abilities.

*Rates refer to subscriptions sent post free to UK addresses.Overseas rates on request.

Please commence my subscription to .....
with the ..... issue. I enclose a cheque/money order for
£..... made payable to ARGUS SPECIALIST PUBLICATIONS
or debit £ ..... from my Access Mastercard/Barclaycard Visa No.

☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ Valid from ..... to .....

Signature .....
Address .....
.....
.....

Cut out and send this form with your remittance to:
The Subscription Manager, Argus Specialist Publications, Argus House, Boundary Way, Hemel Hempstead, Herts. HP2 7ST

WV/6

Quick Merge C64/C128

A program to allow the user to have the facility for a true merge

By Adrian Millett

Anyone who has undertaken any amount of BASIC programming will probably appreciate how useful a MERGE command can be. For instance, the wise programmer keeps a library of standard subroutines and merges them into the body of his program as and when the need arises. While some toolkits fill this gap with a MERGE command, many only provide the inferior APPEND command, which merely tacks one program onto the end of another, regardless of line numbering.

Yet, if you did but know it, the 64/128 has the ability to MERGE two programs (and properly interleaf their line numbers) built in! This flexible technique has a number of other advantages, as I hope will come apparent.

The first step, for both the 128 and the 64, is to load into memory the program from which you wish to take some code, and turn it into a sequential file on your disk/tape with the following direct command:-

OPEN 1, 8, 8, "TEMP. PROG. NAME, S, W": CMD 1: LIST

Or for tape users:-

OPEN 1,1,1: CMD 1: LIST

When the cursor re-appears, type:-

PRINT#1: CLOSE 1

An advantage of this technique is that you can be selective about the bits you wish to merge. For example, you could have used LIST 10000-10999 in the above command.

Now load the main program that you wish to merge with what you have just saved. You are now ready to merge, but first, an explanation of the method involved is called for.

The key routine used here is the Kernal LISTEN function at 65478. If you call this machine code routine with the number of an open file held in the X register, the computer will take all further input from that file. If you do this when you are in Basic, and the file contains suitably numbered Basic program lines, the computer will input those lines as though they were typed in. The source doesn't have to be a disk or tape file:- you may equally well use an RS232 channel as your input.

The use of LISTEN for merging is much more straight forward for the 128 than it is for the 64, so I shall describe the 128 method first. So, in 128 mode, clear the screen and type the following:-

OPEN 1, 8, 8, "TEMP. PROG. NAME, S, R": SYS 65478, 0, 1 or:-

OPEN 1,1,0: SYS 65478,0, 1 for tape.

The disk will now start up and the merge will commence. if "TEMP. PROG. NAME" is very short the file is read from the disk buffer without the drive starting. Eventually the disk or tape will stop, the screen will display either SYNTAX ERROR or an OUT OF DATA ERROR and the cursor will return. Now type CLOSE 1, and you will have the merged program sitting in memory.

The method for the CBM-64 is a bit more fiddly. This is because when a program line is entered in Basic 2.0, the computer closes all open files, so that if you tried a simple adaptation of the 128 method the computer would merge the first line of the file, close the file down and come to an abrupt halt. However, with a little cunning, this problem is not insuperable. The zero page location 152 on the CMB-64 holds the current number of files open. All that Basic 2.0 does when you enter a line is to set this location to zero. So, as long as our input-file is the first in the table of open files, poking 152 with 1 will fool the machine into thinking the file is still open. That, along with a bit of jiggery-pokery to re-enter the poke after each line that is merged, is our basic strategy.

hence, to merge our file, type the following:-

CLR: OPEN 1, 8, 8, "TEMP. PROG. NAME, S, R" or:-

CLR: OPEN 1, 1, 0 for tape.

Now clear the screen and type the following all on one line, ON THE TOP LINE:-

IF ST=0 THEN POKE 152,1:POKE 198,2:POKE 631,19:POKE 632,13:POKE 781,1: SYS65478

This is where you disappear for a cup of tea while your 1541 or tape grinds away, unless you are merging a shortish program or you own a Dolphin Dos, Shark system or similar. As with the 128 merge, if its a very short program, the file is merged without the disk starting up.

When the merge has finished, you will get the cursor back with a SYNTAX ERROR or an OUT OF DATA ERROR. Type CLOSE 1, and your fully merged program is now in memory, ready to save.

While the above methods are not ANSI approved, they are extremely convenient and have certainly saved me a lot of time. you can use this technique to merge suitable SEQ files from other sources, not just one created with a LIST as above. For instance, if you manage to port over a Basic program from CPM M-Basic, IBM GW-Basic or even Amiga Basic (if you add line numbers), and convert the code to PETSCII, you can use the merge method to turn the SEQ file into a CBM-Basic PRG file. Another good idea is to build up a library of useful subroutines in SEQ format. As a start you will find "LINPUT.40000", "DIR.41000" and "AQSK.42000" supplied here.

These routines, although scarcely revolutionary, are good workhorses and will run on the 64, 128, +4 or any PET based commodore machine. They all use the CTRL-C code for abort, but this can be changed easily if you wish.

"LINPUT.40000" is a protected string-input routine, for use instead of the normal error-prone CBM INPUT function, for use instead of the normal error-prone CBM INPUT function.

You need to supply a maximum input length in the variable NIN, and the routine returns the user input in IN$ and the last key hit in CKY, which you can subsequently test for an ABORT.

"DIR.41000" is a universal directory display routine for those machines without a DIRECTORY command. You can supply a wildcard template in A$. If you do not want a template, you must set A$ to null before calling. The variable QE is set to non-zero on return if there is an error while reading the directory. The subsidiary function to check the error channel, at line 41500, is also of general use.

"ASK.42000" is a routine to ask the user for a single-key response to a question. The variable X returns a value corresponding to the response, 1 for the 1st option, 2 for the 2nd, and so on. X is set to zero on an abort. The routine at 42000 is set to ask for (y) or (n) as a fixed question, whereas the routine at 42100 is general purpose, with the user supplying the option characters in B$.

Back to the merge function and a word of warning:- the 128 method doesn't specifically test for an end-of-file whilst its merging. it relies on the presence of a "READY." statement at the end of the SEQ file to generate an error, which "Unlistens" the file. This is OK when merging LISTed files, but if the file is from another source, and this "READY" statement isn't present, the 128 will go into an endless loop after merging. However, if you hit RUN-STOP/RESTORE, you will still find that the file has been merged successfully.

CBM-128 users can automate their MERGE command into a function key, with this command:-

```
KEY 8, "OPEN 1, 8, 8, "+CHR$(34)+",
S, R"+CHR$ (34) + ":SYS 65478, 0,
1[CRSR-LEFT*19]"+CHR$ (27)+CHR$
(65)
```

or for Tape users:-

```
KEY 8, "OEN 1, 1, 0: SYS 65478, 0, 1"
```

When the disk version is executed, insert mode is switched on and the cursor is positioned ready for you to type the filename followed by RETURN. When the merge is complete, type CLOSE 1 as usual.

The complimentary sequential save command, equivalent to the SAVE "FILENAME", A in Amiga/IBM basic, can also be automated thus:-

```
KEY 6, "OPEN 1, 8, 8, "+CHR$ (34) +",
S, W" ≠CHR$ (34) + ":CMD 1: LIST:
PRINT#1," + CHR$ (34) + " READY. "+
CHR$ (34) + ":CLOSE 1"+ CHR$ (141)
+ CHR$ (9) + "[CRSR-UP] [CRSR-
RGHT*4] " + CHR$ (27) + CHR$ (65)
```

Or for tape users:-

```
Key 6, "OPEN 1, 1, 1: CMD 1: LIST:
PRINT#1, "+CHR$(34) + "READY."
CHR$ (34) +":CLOSE 1"
```

These functions work in a similar way to the MERGE command keys. Remember that spacing is critical on all these KEY definitions, and that they are typed in ALL ON ONE LINE.

And finally, for the owners of Commodore orphan machines, the 128 merge MAY work on the Plus-4/C16 and the CBM-64 method SHOULD work on the VIC-20. Unfortunately I don't think I can be of help to KIM-1 users...

Techno-Info

Jason Finch gets out the books and crystal ball to answer more of your problems

Dear CDU,
I have been purchasing CDU since issue number one and have found the disk programs to be very good. I am interested more in "serious" programs but I also like to see the quality of the games. On April's disk I found that the Texas demos and Bar Prompts programs contained some very practical and useful ideas. The utility programs published in your magazine are also of a high quality. Congratulations, but please NO more "Programmer's Diary". This article is a waste of magazine space and full of errors – he is talking about self-hypnosis and not meditation and he needs to understand the difference. I would have much rather had the technical details from Mike Holmes Auto, Delete and Renum. All I can say, I'm glad I'm not a programmer! I would probably award your magazine a nine out of ten because on the whole it is excellent. On page eight of April's issue, in Techno Info, you mention a company FSSL. I would not, personally, do any further business with FSSL. They offer a very indifferent service and take ages to reply. They charged my credit card and did not send the goods!! Buy GEOS direct from Berkeley Softworks. I now buy books and programs from Software Support International, 2700 NE Andresen Road, Vancouver, WA 98661, USA. Your order is received within two weeks. An excellent company to do business with!!
N.K. Taylor, Bournemouth.

Dear Mr. Taylor,
Thankyou very much for your letter. We need our readers to write and tell us what they are up to and what they think of various things in CDU and the service they get from companies. I think that nine out of ten is fair – it gives us a little room for improve-

ment. Peoples' opinions on different matters vary greatly and I am sorry that you do not enjoy "Programmer's Diary" by Andy Partridge. I am sure there would be people who would complain if the diary was omitted to make room for the technical details. We try our best to please everyone which just isn't that simple. It is a shame about FSSL. I have purchased a wide range of software and hardware from them in the past and although they do usually take a while to reply, none of the products have been defective which is the important thing. There are some people that prefer to keep their money and orders close to their chest in England but thanks very much for the address of Software Support International.

Dear CDU,
Last issue (April) you received a query from 'Michael' who had problems with an incorrectly installed GEOS utility. Since FSSL are the main distributors of GEOS you, quite rightly, gave their number so that he could seek assistance. What you did not consider was the fact that FSSL are in the business of selling software and are not really interested in repairing software which has been corrupted by fair means or foul. but all is not lost. FSSL sell a utility called Maverick for £24.95. Apart from being able to copy virtually anything it will also write GEOS parameters. I solved my problem by copying geoPublish with the disk copier and writing new parameters. I can now use the copy with any version systems disk. It still doesn't alter the fact that the original can only be used with v1.3 and if the Maverick is used for anything other than genuine personal backups, copyright is infringed. One thing is sure, this is probably the only way Michael will be able to solve his problem. Now a quick word of warning. Anyone ordering a HandyScanner64 beware. In the advertising it tells of detailed instructions – one major problem: they are in German. For more than two months I have been badgering them about getting translations but to no avail. At £235 a throw it's not on to have to bumble through on ones own. I'm hoping a bit of publicity may shake them into action. FSSL market very good gear and their after sales service is usually very good and other than the stated problem there are no complaints and I would thoroughly recommend them to anyone.
J.J. Malinowski, Lincolnshire.

Dear Mr. Malinowski,
Firstly I shall cover myself by saying that it was FSSL's Technical Support Service that I recommend when I thought was a perfectly reasonable thing to do for such a query because the program did not need "uncorrupting" as such. But thanks very much for all your information. I am sure that Michael will find it useful. The problem with FSSL is reflected in the first letter so come on FSSL – buck your ideas up. But as you have said, FSSL do market some excellent software and hardware, including a new range of hard drives for the 64. You said you would thoroughly recommend them to anyone. In comparison with the previous letter, it just goes to show that there are two sides to every story and nobody thinks the same.

Dear CDU,
I have noticed with great interest the amount of people who have the same problems with their programs on the disk as I had. I give Wabbit as an example. I read in a past issue that problems are occurring with black screens. I too had this problem until a friend of mine gave me the solution. Previously I had many disks which ran all the programs except maybe one or two on each disk. Now all my programs from CDU are running perfectly well. The solution? Well it may sound silly or it may sound too simple but believe me it has cured all my problems and I hope it will cure yours. Make sure that your tape deck is plugged into the back of the computer. It has given me all the programs that wouldn't work. I don't understand why but take my word for it, it works. I hope this little bit of info will help you and everyone else to enjoy the fabulous programs from the best C64 mag around!
John Benson, Northern Ireland.

Dear John,
I must admit when I first read your letter I thought it sounded a bit like one of these miracle diets – no offence intended, please. Due to my needs I have upgraded to a 128 but still use the old 64 to check things like Wabbit. I tried your solution and unfortunately it did not work, whether I had it just plugged in or with the play button down, the cassette rewinding or anything else I could think of. Perhaps it is some strange quirk with a particular model of tape deck and 64. Still, I would be interested to hear from anyone else who is successful with this method and please state the model of your cassette deck. Thanks for the information.

Dear CDU,
This year I received a second hand C64 for Christmas, as well as the Advanced OCP Art Studio. I wish to print my pictures out on my Commodore MPS802 printer. Unfortunately the printer was suffering from Manualus Lostus when I bought it. This means that I cannot configure the printer for the program. Please could you tell me the answers to the questions in the configuration program.
Duncan Martin, Essex.

Dear Duncan,
I am always loathed to tell people something won't work because I usually end up getting a slap-wrist when someone writes and tells me it will (which I don't mind!). But unfortunately I am almost positive that the program won't output pictures to an MPS802 printer. It will output to an MPS801 and 803 but not an MPS802. Sorry about the bad news.

Dear CDU,
After loading the latest offering from CDU (April) into my trusty 64 I am pleased to say that once again you have come up with another fine selection of demos – especially DELIRIOUS VI which in my opinion is the best so far. Talking of demos, when will this fantastic demo of Andy Partridge's be appearing in the mag?? I really want to see this

one! Having read the magazine I also see that you wish for us readers to come with some useful suggestions. Well – read on... How about some more games reviews than just one or two and what about more competitions? Some hardware reviews maybe and a sales/swaps page for readers to exchange software and so on. These are just a few ideas for you to work by.
Mike Pitches, Plymouth.

Dear Mike,
Hello again! I hope you got the colour rolling effect working that I told you about in the February issue. But that's beside the point. Thanks very much for all your suggestions. But to hear that you can't wait to see Andy's demo! Oh no – what is this world coming to! I must say that I would be disappointed to see more reviews because CDU actually aims NOT to be a games mag, leaving that sort of thing to its sister magazine YC. More reviews would mean less space being devoted to useful programs and of course this wonderful "regular helpline" may get reduced and we wouldn't want that – well, I wouldn't anyway! We shall have a look at the compo situation but there is one running at the moment for a colour printer – perhaps you could try writing your own demo fro that. Hardware reviews are done now as and when available and a sales and swaps page could be a feasible thing if sufficient interest is shown. Once again, thanks for your interest in the magazine.

Dear CDU,
I have two problems with my Commodore 64 which I hope you will help me with. The first problem is when I use YC Writer, a word-processor program from the Commodore Serious Users Guide 1988. When I load the program all is well until I use the quotes when I get an odd effect – ie "(down)c (down) o (down) m (down) m (down)..." I am using a STAR LC10C. The second problem is the TEXTED word-processor in the July/August CDU a little while ago. When I type the address as you see at the top of the page it prints everything side by side. I know it must be my mistake but I don't know what it is. Please could you help me.
T. Eley, London.

Dear Mr. Eley,
The reason, I would think, that you get such strange results is because quote mode is enabled after the quote mark and stays enabled until another is encountered or a carriage return is output. You see the effect in a standard PRINT command. The word-processor must output a control code to the printer before each character – a cursor up (145) for capital letters and a cursor down (17) for lower case is the norm in such circumstances. And because quote mode is enabled these codes appear as their actual characters or as a string indicating what the character is. You will just have to omit any quotes or use two apostrophe marks together. The problem with the TEXTED wordprocessor is that it is set up to print out the text with 160 characters per line and not eighty as expected. It therefore prints pairs of lines next to each other but if you are printing in 160 columns you don't notice this, nor do you notice it if you are printing eighty characters per line (such as on an MPS801) but with full length lines. There is no easy way to rectify the updated TEXTED that you are using because it has been compiled but the original should be changed in line 470 from PRINT#1, TEXT$(X); "(space)";:... to read PRINT#1, TEXT$(X);.... In other words you must omit the two semicolons, quote marks and space immediately following the statement. This means that only one eighty column line is printed and not two next to each other. This should cure the problem. I hope I have been of some assistance.

Dear CDU,
I notice from your latest issues that you are supplying GEOS fonts for GEOS users. I purchased the Advanced Art Studio and with it received a mouse. This contained a disk with GEOS on. However, I cannot find out how to load the GEOS system!!! The disk is double sided with demos on one side which I can load and they work, and what I believe is the GEOS system on the other side. The side which I cannot load is marked "GEOS V1.3 UPGRADE". The directory of this side shows four programs, all upgrades of some kind, one of which is protected. They are also USR files so I have no idea how to load them as I am new to disk drives. Please could you tell me if this truly is the GEOS system and if so can it be loaded? I would very much appreciate a reply as I have had this problem for a couple of months now!
Mark Hobbs, Dunstable.

Dear Mark,
I am afraid that the programs on the disk will not produce any results and cannot be loaded either until you have the main GEOS system disk which must be purchased separately. This is not any part of the main new operating system of GEOS. Should you decide to purchase GEOS you probably won't need the files either because they upgrade a present system to v1.3 and it is v2.0 that is now already available. Once again, the main GEOS system is not on that disk.

Dear CDU,
Just one quick query. If I purchased the GEOS geoWrite word-processor would I need any other software and so the extra fonts work on the MPS803. Thank you for an excellent magazine.
L. Walls, The Isle of Wight.

Dear Mr. Walls,
To use the geoWrite word-processor all you need is the system disk. Usually contained in this package is the word-processor and so that is the only software you require. The fonts will all work on the MPS803, no matter what they are, how large they are or how you decide to print them. This is because they are actually graphically based – it is like taking a high-resolution picture and dumping it to your printer.

Dear CDU,
With reference to my letter to you regarding SEQ files and using these in my programs for the 1520 plotter. Thanking you very much for the great help given by Jason Finch of CDU, Techno Info and the pro-

grams supplied. It is nice to know that a mag cares about its readers in this age.
Ken Taylor, Plymouth.

Dear Ken,
I thank you very much for acknowledging my programs. I am very glad that I was able to help you. "but who is this person?" I hear everyone shout. Well Mr. Taylor's query did not appear in the mag – I replied direct. Obviously all the queries cannot be used in the mag and so a few have star treatment! Unfortunately it is not always possible to reply to everyones letters but rest assured – every single one is read. Thanks once again - it is great when everything is sorted out.

Dear CDU
Thanks for your letter of the 28/ April which we have read with interest, are problems using the reset button, which you have attributed to incompatibility with 'older' models.

We did see the letter from Mr Booth published in CDU and do recall him calling us. May I say that we were always courteous to him, and never tried to fob him off. The honest truth was, and is, that his problem was so unusual that KCS the manufacturer had just never heard of it. Mr Booth never asked us if he could return the Cartridge, and we presume he finds the other features very useful. None of the other people have contacted us either.

What I would like to do is set out some facts which will clarify the situation.
1) This Cartridge was developed in 1985/86 and the software has never been changed since then, (expect for changing the logo date on the screen from 1985 to 1989). In other words it was developed on 'older 64's and not with the latest models. Commodore-Holland supplied KCS with various older and current models available at the time, and with later ones as they came up, for testing purposes.

Hence we find it very difficult to accept a sweeping statement that A) the Cartridge does not work on older models and B) that it was not tested sufficiently. (An independent magazine review was done totally on a 1984 model.)
2) Our company Bitcon Devices Ltd were the ones who initially supplied KCS with the memory-ram chip used in the Cartridge (Mororola 6810) back in 1985/86 so we know that it is the absolute truth that 110,000 have now been sold. We had personally spoken to Dutch and German distributors before taking on this product for UK distribution, and none of them ever mentioned that they had any such problems with the product. In fact they considered it a very saleable item with extremely few returns. This has been confirmed by our UK distributors too. (Ask Lazer Distribution for example who just this week commented favourably to us on it).

We honestly do not know with certainty the reason for Mr Booths problem, but we can say with certainty that it is not for any lack of testing or because of negligence on KCS part.

Maybe these customers have grey-import USA models with slight incompatibilities. We have had, albeit very occassionally, people complaining that they can't get their Cartridge to work. Not that the reset would not work, but that they could not load etc., and when we have tested it it has been fine. Nevertheless, we have invariably replaced it with another and their problem was solved. Nothing is compatible with everything! You will surely agree.

To recap, it is not a widespread problem at all, and I even wonder if those customers who have written in with reset problems actually mean the same thing as Mr Booth, as a few people are under the misunderstanding that this is a reset cartridge allowing one to add pokes and re-enter the game. This cartridge does not allow it and they are losing their program when trying. As you know our reset only means to clear the memory. In conclusion, after what we have stated above, it would be very wrong if the impression was given in CDU that our Cartridge was incompatible with older C64's, which is simply not true. And if people have problems, we are always prepared to listen.

The above letter was written by Mr. J. Kaufman of BDL Ltd in reply to a number of queries we had received concerning the "Power Cartridge". CDU would like to thank BDL for their swift and helpful reply. May I just add that clearing the connections on the computer should also help those that have experienced problems.
P.A.Eves – EDITOR

Tip of the Month

I am really getting annoyed now – we still haven't had many of you writing in with your tips. Obviously the thought of fame and endless fortune does not appeal. So I shall have to give you another little bit of useless info seeing as how none of you are bothering! It is a small but potentially useful routine written in machine code but to be used from a BASIC program. It is a selective GOTO statement to allow you to jump to lines, depending upon a variable. Now I know that you have the ON...GOTO statement but that is not the same. Suppose you have a variable L which represents the level you are in one a game and X is a room with that level. You could want to access a particular line to draw the level such as GOTO L*1000+X*50. With a normal GOTO this is not possible but enter the DATA below to the start of your program as shown and you will be able to enter SYS679 (L*1000+X*50) with absolute ease! Use it where you like and when you like but remember the brackets. Another example – GOTO (Z*250) where Z could be a verb number in an adventure. Well here goes, this should be the first line of your program: 5 FOR N=0 TO 8: READ X: POKE 679+N,X: NEXT: DATA 32, 241, 174, 32, 183, 76, 163, 168. The machine code instructions are JSR $AEF1, JSR $B7F7 and JMP $A8A3.

Well, that's it for this month! If you have any programming problems or queries then please write to Techno Info, CDU, Argus House, Boundary Way, Hemel Hempstead, Herts. HP2 7ST. That is also the address to which you should send information if you want to be the first to share your knowledge with the other readers. of course, the social esteem gained by such an event far outweighs any amount of cash that could be offered!! See you all again next month.

£1000 REWARD

FOR INFORMATION LEADING TO A PROSECUTION & CONVICTION

THIS CAMPAIGN IS ORGANISED BY

ELSPA

EUROPEAN LEISURE SOFTWARE PUBLISHERS ASSOCIATION

ANY INFORMATION ON PIRACY SHOULD BE PASSED TO F.A.S.T. (THE FEDERATION AGAINST SOFTWARE THEFT)

TELEPHONE 071-497 8973

PIRACY IS THEFT

ARCADIA

Julian Woodford explores the connotations derived from the marriage of film fantasy and computer play

by Julian Woodford

If you were watching T.V. late one evening, just after Christmas, you might have noticed an odd and very short film called Arcadia. Its title is Greek, but it is also present in Elizabethan love prose, and a term indicative of a pastoral ideal, full of primitiveness, music making and dancing. But it also has the meaning of the spaciousness of an 'arcade' derived from Georgian shopping centres, French and latinized open pilasters rather than Wardour Street amusement arcades. Arcadia worked with both approaches, not in this semi-intellectual way but as director Paul Bamborough says "as a means of entertaining and of having fun".

Amusement Arcades are an almost omni-present feature of seaside and town entertainment. An unimagineable number of coin-operated machines proliferate from Soho down to Brighton. Some people must have seen them all: Who? Well, in this instance a young man called Gavin who has to battle through a definitely non-acradian world to reach his own amusement arcade.

That is the basic plot to this ten minutes, or so, film. It centres not exclusively on the life of Gavin. He lives in a world noticeably similar to our's but uniquely different: he has a mother and father, his mother gets him breakfast, she wears mum type clothes, he does son type things but she carries an automatic rifle and he sleeps with a pistol. "In Arcadia you trust no one".

After breakfast Gavin announces that he's going out into a world seething with animated monsters, wreaked by explosions and full of equally mis-

In Arcadia you trust no one.

The world is under seige and for Gavin's mum even breakfast could be a trap. When Gavin announces he's going out his parents suspect the worst. They're sure he's becoming a delinquent. But Arcadie isn't like our world and you mightn't suspect what kind of delinquent he is. And Gavin himself has no ideas of what's in store for him...

trustful people. his parents know that he's done it fearlessly facing death to get to his arcade bunker to play a very testing coin-up. Greeted there by his nervous friends he proceeds to play. But what happens turns our idea of how coin-ups, and how computer shoot-em-ups works on its head.

If you were to apply a typical level to level computer game strategy to Gavin's activity you might get the following:

Level 1.
Wake up, get up, avoid being shot by mum, go downstairs, avoid being shot by dad, tell them you're going outside, avoid being shot by both of them.
Level 2.
Go outside, avoid being shot by the two types of animated nasties using your gun, craters, walls and ceilings as protection. Cross the open spaces to the arcade hall.
Level 3.
Enter the arcade hall, avoid being shot by your 'friends' Go the coin-op. Begin to play the peaceful coin-up.
Level 4.

Playing the peace game – do not shoot people in the game, whatever you do. If you bump into a woman apologise to her, if you get into someone's way apologise again, deprecate. Even when you get wheel clamped, receive a parking ticket, and get verbal abuse then be differing.
Level 5.
The penultimate Level. You not only try to defeat the peace game but you also try to defeat the real world and transcend the isolation, or 'entertainment' of just playing a game.
When the coin-op self destructs do not shoot it either with your real gun or your computer gun.
Level 6.
The Final Level – Having learned peace by not shooting the machine to bits, when it won't let you play anymore, return home to mum. Tell her you love her and see what happens.

If you're as tired as I am of reading all the novellas, short stories, blub and general tittle-tattle that seems to come with all computer games then perhaps you won't have got this far with Arcadia. But if you have then you get 10 out of 10 for a gutsy reader. The final level is one of the areas where Arcadia succeeds most the inversion of expected events: the aim of the game is non-violence; you win by not killing.

In that sense it imparts a new morality to the world of computer games: I defy you to find a game that has a similar peace keeping structure. But 'morality' is not even on the short list of why the film was made.

It's writers Phil Austin and Derek Hayes, of Animation City, and Director Paul Bamborough, had the basic idea of Arcadia on the back-burner for quite a while. In the early 80's Animation City did a cell animated fantasy sequence for a semi-documentary called Arcade Attack.

They worked on the animation as a fictorial addendum to the documentary's analysis of arcade games and the age gap: dads played pinball, sons now played space invaders. So the sequence had living rooms and houses invaded by the young ghost of a new generation – space invaders. "What kind of video games would the old teddy's have played' was the major directive behind what they tried to achieve out of that particular film. Arcade Attack slotted nicely into the atmosphere of change about the arcade machine and the arrival of the home computer.

It's not rare in film making to have a number of different ideas developing at the same time – it's probably a necessity. So even though all three of them were working on short pieces from pop commercials to ads and cartoons the desire to make a film along the lines of Arcadia was continually there.

In the end the film that came out, in conjunction with Bristol Screen Finance and Channel four, was an 11 minute film that all of them would have liked to see: "lots of people running around, noise, explosions and fun".

No computer graphics were actually used, it was all hand drawn cell animation matted onto the film.

In general terms, the technical quality of the film is magnificent, both in its scripting and animation sequences; with arcade ambience added by animated explosions matted on later.

The Game Plan

Let this utility work out the coordinates of sprites and background objects necessary for collision detection with ease.

By Mike Benn

Imagine the situation; you are creating a mega arcade game where your character has to battle his way through dealing with countless problems. It has all the makings of a great hit and clearly you want to produce the finished result as soon as possible. When the game play is worked out and the graphics completed there comes a point when you have to marry up the action with the background. Take for example a hero battling his or her way across your newly created background. The characters progress around the screen will be hindered by many dangers, possible items to collect or restricted access to parts of your backdrop. Problems can be created totally randomly or, more interestingly, when a point reaches a particular part of the background. Perhaps our hero is wandering through a game and discovers an alligator filled river to cross the problem for the player is how to cross the river and avoid becoming the gators next meal. In this case the programmer having designed the scrolling background has to know precisely where the river is on the backdrop. The position of the gators (ie sprites) on the screen and the limits of their movement. The player should have some chance of getting past and limits of movement need to be calculated before they can be included in the program.

Testing the movement and discovering the co-ordinates of sprite positions can often mean running the games program on a trial and error basis before the game play is satisfactory. If the game includes a scrolling oversized backdrop it can often be impossible to determine where the river is in our example if it were part of a larger background.

As you may have guessed GAME PLAN is a utility program to help you out with such problems. The program gives the X,Y coordinates for any sprite position on the screen so you can determine the positions and limits of any sprite movements. The program allows for scrolling backgrounds mapping out the X,Y coordinates as with the sprites. You simply jot them down for inclusion in your master program.

Using Game Plan

The program is activated by typing SYS49152 and you are presented with the LOAD screen. This screen gives you the following options:-

LOAD SPRITES
LOAD CHARACTERS (USER DEFINED)
LOAD MAP (BACKDROP)

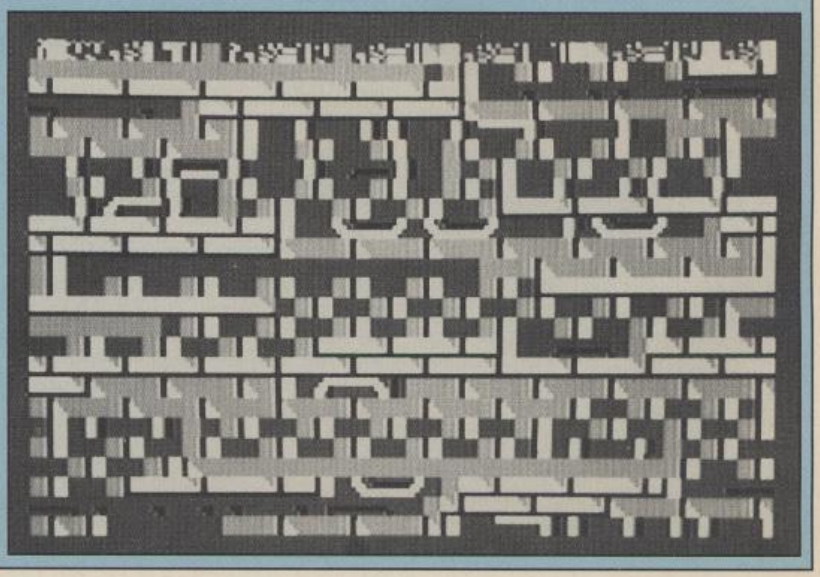
Sprites are loaded in at $2000 (sprite definition $80)
Characters are loaded at $0800
The Map is loaded in at $4000. The map size has the following maximum limits:-
WIDTH 256 CHARACTERS * HEIGHT 128 CHARACTERS The minimum can be as small as 1*1.

When loading is completed you are ready for the display screen. This shows the coordinates at the bottom of the screen for both sprite and background movements. The joystick (Port B) controls the sprite movement with the function keys taking care of the backdrop. The manipulation of such delights as colour, modes, sprite priority and sprite expansion are controlled from the keyboard (see table).

A few points about the output data. All information is given in HEX; this is to partially save space but, as most arcade games are written in machine code it's more useful. As the sprite moves across the screen and into the area of the MSB (Most significant Bit) the data line prints an 'M' while it remains in this portion of the screen.

The program needs to know when working with non standard backgrounds (ie larger or smaller than 40 characters wide) how wide your background is otherwise you will find your design scrambled. The output for the width control is printed in the X coordinate of the backdrop map. There should be no conflict with the X position as moving the screen in the X axis will return the output to normal use.

If you choose not to load any sprites the program provides two definitions in both modes at 160 and 161. All will become clear when you run the program I promise. To start, you off load both the character and map demos. When you are on the display screen alter the map width to $40 and scroll the backdrop around.

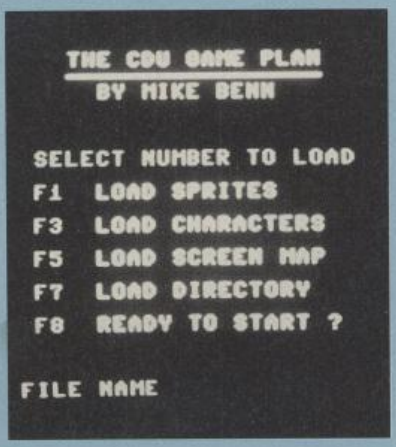


Key Table

F1	LOAD SPRITES
F3	LOAD CHARACTERS
F5	LOAD MAP
F7	DISK DIRECTORY
F8	DISPLAY SCREEN

Display Screen

F1	MOVE MAP RIGHT
F2	MOVE MAP LEFT
F3	MOVE MAP DOWN
F4	MOVE MAP UP
F5	INCREASE MAP WIDTH
F6	DECREASE MAP WIDTH
F7	INCREASE SPRITE DEF
F8	DECREASE SPRITE DEF
P	SPRITE PRIORITY ON
SHIFT P	SPRITE PRIORITY OFF
	SPRITE Y EXPAND ON
SHIFT	SPRITE Y EXPAND OFF
	SPRITE X EXPAND ON
SHIFT	SPRITE X EXPAND OFF
M	SPRITE MULTICOLOUR MODE
SHIFT M	B/GROUND MULTICOLOUR MODE
H	SPRITE HIRES MODE
SHIFT H	B/GROUND HIRES MODE
Q	QUIT



0	INC. SPRITE COLOUR
SHIFT 0	DEC. SPRITE COLOUR
1	INC. SPRITE MUL/COLOUR 0
SHIFT 1	DEC. SPRITE MUL/COLOUR 0
2	INC. SPRITE MUL/COLOUR 1
SHIFT 2	DEC. SPRITE MUL/COLOUR 1
3	INC. CHARACTER COLOUR
SHIFT 3	DEC. CHARACTER COLOUR
4	INC. B/GROUND COLOUR
SHIFT 4	DEC. B/GROUND COLOUR
5	INC. B/GROUND MUL/COLOUR 0
SHIFT 5	DEC. B/GROUND MUL/COLOUR 0
6	INC. B/GROUND MUL/COLOUR 1
SHIFT 6	DEC. B/GROUND MUL/COLOUR 1
7	INC. BORDER COLOUR
SHIFT 7	DEC. BORDER COLOUR

JOYSTICK TO MOVE SPRITE

Diamond Bytes

Diamonds are a girls best friend but are they the programmers? Your roving reporter travels to Rotherham to find out

By S. Wickham

In these days of games, games and more games, it is nice to see there are a few software companies still dedicated to the serious computer user. Unfortunately, the number of companies producing 'Utility' and 'Applications' packages for the C64 and C128 are dwindling, at least in this country. It is therefore with great pleasure that I can write about a new company dedicated to the non-games computer user. The name of the company is 'Diamond Bytes' and they are based in Rotherham, Yorkshire. I decided to drop in on the operations room during this Easter week-end. I purchased my British Rail super saver and settled back for a pleasant trip. On arrival in Rotherham I hailed a cab and ten minutes later I was sitting amidst a mounting of disks, computers, printers and duplicators.

Before I set off, I did my homework and discovered as much as I could about the company. They have been in the business for some two years now, their forté being Duplicating. They have been slowly growing and gaining the reputation as a reliable and convivial company. Everyone I have met that has dealt with them in the past always had a good word to say about them. (Just like they do on This Is Your Life) I thought this was too good to be true, so I prepared a set of suitable questions, and with pen and pad at the ready I settled down for an interesting interview.

CDU;
What made you decide to branch out into the word of software producers as well as being disk duplicators?

DB;
We started off originally just earning ourselves a bit of extra so that we could enjoy a few luxuries in life. To our dismay we not only became fairly successful, but we also started to learn a lot about the software industry as a whole. Like most people, we decided that we could do a lot better than we were at that time.

CDU;
These days most software producers concentrate on the entertainment side of the market. Why did you opt to go for the serious user instead?

DB;
The software industry is a very competitive business, just like the Pop and Video industry. In order to compete against the big boys like Activision, Ocean and such like you would need millions stashed away somewhere, let alone an army of programmers at your beck and call. A quick look at the list of software producers told us that not many people are producing serious stuff these days. So why not capitalise on this fact and start producing our own.

CDU;
You have chosen the name 'Diamond Bytes', does the name suggest anything about your company, or is it just because it sounds nice.

DB;
This is a pretty good question! Most companies like to portray what they do somewhere in the name. We wanted to shy away from the obvious names like 'Software, Computers etc', the second word 'Bytes' is sufficient to let you know that we deal with computer software. The first word 'Diamond' will hopefully portray our standard of products. That is to say they will be long lasting, very popular and will always set a shining example of quality software.

CDU;
Just how do you propose to achieve your aims that you have set yourself?

DB;
Basically we intend to only produce top quality software that serves a real purpose for it's user. Initially we will be commissioning programs by individual programmers. However, once we start to take off we hope to form a team of in-house programmers specialising in writing utility software. As you are no doubt aware, projects written by a team tend to be that much more professional than individual efforts. Add the fact that all our programmers will be specialists in the field of utility writing, the formula cannot fail.

CDU;
Do you intend to produce only utility software and no entertainment software at all, and on what formats are you producing for?

DB;
We will be producing some games orientated stuff once we become more established. But as said earlier, it is not our aim to be yet another games producer. We firmly believe that there is still a market for utility software in this country. We will be producing software for all Commodore formats. That is to say C64, C128, Plus4 and Amiga. Whether or not we expand into Atari, Amstrad and PC formats will remain to be seen.

CDU;
A lot of computer owners have one basic criticism about software, namely the price. How do you intend to price your products?

DB;
We agree with the majority of users on this one. The price of software is generally quite high. A lot of the reason for this is the large amounts of money spent on software protection. We do not intend protecting our disks at all. We believe that these days, contrary to popular belief, most people prefer to have original software. Providing the software is nicely packaged and presented, and does the job it is supposed to do, then most people will buy the original. The outcome of this is that our products will be marketed at a reasonable price, both for the users and for the programmers.

CDU;
How many products have you got lined up for your launch, which I believe is the end of May?

DB;
So far we have got 10 products we are working on. The first of these is a machine code programmers dream. 'CODEMASTER' as it is called, is an extensive package offering amongst other things; A de-bugger, analyser, assembler and patch editor. The second is a superb utility called 'POWERTOOLS'. This package will offer A program linker, A program compactor, A program scrambler and various other routines. There are also 4 utilities for Plus4 users being made ready. The Amiga users are not left in the cold either, we have a very nice art package coming out shortly. (I cannot say too much about this at the moment).

CDU;
I would like to thank you for taking time to see me and answer a few questions. We will certainly keep an eye out for your products on the scene.

DB;
Thank CDU for asking us to contribute to your magazine pages. We certainly hope that we can fulfill our dreams and cater for the more serious computer user amongst us. We feel that there is an ever increasing lack of utility programs being made for the CBM range of computers. The European market is streets ahead of us on this issue. Let's put the UK back on the map as the leading utility software producer.

Character Designer

Redesign the Commodore 64 character set with this easy to use utility

By Tim Hanson

The Commodore 64 character set can be altered, but with great difficulty, to suit your own personal needs. Objects in games can be more identifiable. Text characters can be redesigned to give new and more welcoming fonts.

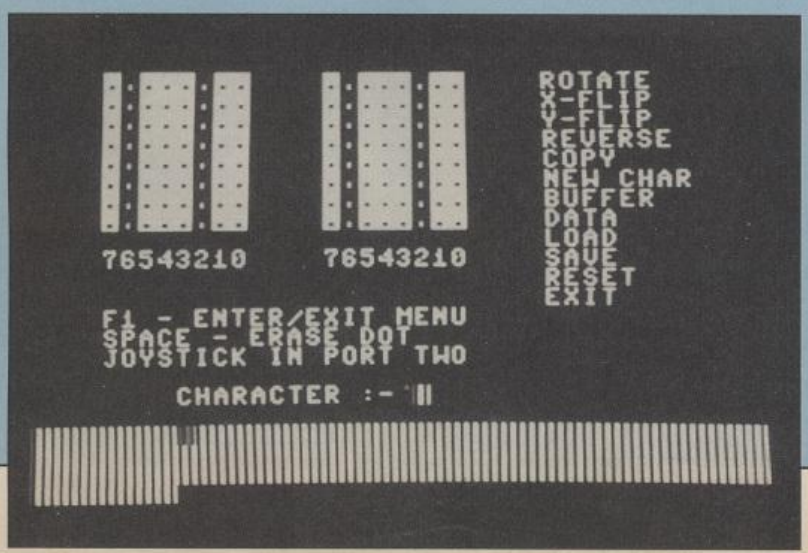
Redesigning characters normally involves many laborious binary to decimal calculations. With "Character Designer," characters can be redefined and edited quickly and easily. Character Designer is written in machine language for maximum speed. To use Character Designer select it from the menu or alternatively type the following:-

LOAD "CHAR. DESIGNER", device (RETURN)
RUN (RETURN)

Character Designer can be loaded and copied like a normal BASIC program.

Designing Characters

After running character Designer, the computer prompts you for the source of the characters. The program makes alterations to characters at 12288 ($3000). If there is a character set at this location which you wish to alter then press "F1." Pressing "F2" will copy the standard character set from ROM to 12288. Character Designer will now ask for the character you wish to alter/edit. Commodore and shift keys can be used for the selection of graphic characters. Only a printable character will be accepted. (The program will not accept cursor keys, function keys etc). After selecting a character, Character Designer will ask for the type of character from the selection of normal, reverse, lower case, and reverse lower case. After entering this information, Character Designer will display the character, the screen code, and the address of the character information. Press any key to enter the editing screen.



Editing characters

The first thing you will notice about the editing screen is that it has two grids. Characters can only be edited on the left grid. The right grid acts as a buffer which can be copied or exchanged with the editing grid. On the right of the editing screen is a menu. This can be entered and exited by pressing "F1." Displayed at the bottom is all of the characters in the current set. Just above the characters, the character currently being designed on the editing grid is displayed at actual size. To edit characters, use a joystick in port two to move the flashing square around the grid. The fire button will set pixels and the space bar will erase pixels. When "F1" is pressed, the menu is activated. To select an option, move the selection bar with the joystick and press fire. Selecting the following options will perform the subsequent operation:-

ROTATE – rotates the character-clockwise 90 degrees.
X-FLIP – turns the character upside down.
Y-FLIP – turns the character backwards.
REVERSE – creates a negative image of the character.
COPY – copies the buffer grid to the editing grid.
NEW CHAR – changes the character in the editing and buffer grid.
BUFFER – only changes the character in the buffer grid.
DATA – displays character data in hexidecimal form.
LOAD – loads a character set into $3000 to $4000.
SAVE – saves the complete character set from $3000 to $4000.
RESET – restarts Character Designer.
EXIT – exits Character Designer.

Accompanying these menu options, various keys access other function.

```
CHARACTER DESIGNER BY TIM HANSON

1) CHARACTERS FROM $3000
2) DOWNLOAD ROM CHARACTERS

PLEASE SELECT CHARACTER - H

1) NORMAL, 2) REVERSE, 3) LOWER CASE OR
4) REVERSE LOWER

CHARACTER H  SCREEN $08  ADDRESS $3040
      PRESS ANY KEY TO CONTINUE
```

B – changes buffer character.
C – copies buffer character to editing grid.
D – scrolls character down
E – exchanges buffer and editing grid.
L – scrolls character left.
N – changes only the editing character.
R – scrolls character right.
U – scrolls character up.

Using the Characters

After designing a special character set and saving it, it's time to incorporate the characters into your own programs. All that is required is to load the character set and activate them. To load them form a basic program a line like this would have to be one of the first lines in your program:

```
10 A=A+1: IF A=1 THEN LOAD "file-
name", device,1
```

To activate the character set, include a line like this where they are needed:

```
20 POKE 53272,29
```

How It Works

Character Designer uses a high resolution screen for editing. The purpose of this was to be able to display the edited characters without destroying or changing the characters on the editing screen. The prompting of a file name for loading or saving is the only time a text screen is used. To place text characters on the high resolution screen, Character Designer prints the text to the regular text screen and calls a subroutine to convert this screen to one of high resolution. If necessary, direct access's to the high resolution screen are made to store images of the edited character.

Adventure Writing

Jason Finch continues his tuition on writing your own adventures

In the first part of this tutorial into writing your own adventure programs I explained a few techniques for the displaying of graphics, should you wish to include them in your adventure. On this issue's disk there is a file called "PIC1". This is a picture file that will eventually be used in the final example adventure. Further ones will be given in future issues. The pictures were created by Doug Sneddon (thanks very much!!!) and I have condensed them slightly to allow more disk space. The data itself is not physically changed but each picture is only thirty characters horizontally and all the data for each line is pushed together end to end. There is nothing really complex about this method and it is only specific to this case and therefore I feel that explaining it in detail will be pointless. All unused bytes are simply eliminated and then a routine is used to display everything centrally on the screen. This will be provided later together with a routine to produce a "split-display" with the bitmap at the top and text at the bottom. Now, though, I shall explain a few techniques for the storage of information and the possible advantages and disadvantages of programming it in BASIC as opposed to machine code and for writing the adventure for tape or disk.

If you are planning on creating an adventure that will not consume a great amount of memory, such as one with a large number of locations but with short text descriptions or vice versa, and want to write it in BASIC then all information can be loaded in one go and stored in string variables. However, if your planned adventure is likely to contain a large amount of text, for example due to it having a great number of locations with long and detailed descriptions, then it may be worthwhile storing them on the disk and then loading each as and when required. This will give more programming space for other aspects of the adventure. There are two main ways of dong this.

The simplest method would be to store each as its own sequential file and give each the filename LOC1, LOC2 and so on. However, you must be careful if you want to use the INPUT# command because the longest string that this will accept is 80 characters. The best way here is to use the GET# command build up a string from separate characters, checking for a null string to signify the end. However you could instead use the drive's direct access commands such as Block-Read, Block-Write and Block-Allocate to create and read the descriptions directly to and from the individual sectors and tracks of the disk. This may be more professional although it is more difficult to program and tabs must be kept on where each description starts and remember to take care that the disk is not validated or the Block Allocation Map on the disk will be altered and you may end up accidentally erasing some of your data if you write something else to that disk afterwards.

The ability to store and recall information quite quickly and easily is of course the main advantage of creating a disk-based adventure rather than one for cassette. If you plan to use detailed multicolour graphics then these can be stored on disk as well and recalled if and when required. If you want an example of the two forms of text storage then also on this issue's disk you will find the file "AW-TEXT STORAGE". You will need a blank disk handy as the program will format it to ensure that all the blocks are clear. A few descriptions will then be saved to the disk. You can then recall these however you like. List the program – it is fully REMmed so that you can see what is going on. The program procedures are fairly slow because it is written in BASIC.

If you want to produce a first class adventure and are extremely competent with programming in machine code, then the latter language will be far quicker – but a machine code adventure is usually the product of months, if not a couple of years, work. However if you would like to produce an adventure in a reasonably short period of time but still of a fair quality then there is no reason why you shouldn't write in BASIC. Although it is much slower than machine code, and even more so if the drive is in constant use, speed is not necessarily an important factor. If the adventure is slow you could argue that there is more time to think between the entering of commands. I find the best approach for an adventure to be written fairly quickly is to write the bulk in BASIC and to have selected routines in machine code. If you only program in BASIC then don't worry – just remember that with an adventure speed it is not necessarily important.

Now we have considered a few methods for the storage and retrieval of information and so I shall describe very briefly the storyline of Demad, the example adventure, which will be introduced to your properly in the next issue. It is quite a "small" adventure, there being only eleven locations. The whole idea is to talk about and collect items that allow you to overcome certain problems. For example, there is a witch that holds one of the three coins that you must collect in order to finish the adventure. To persuade her to release the coin you must give her something that she may find useful, exactly what though, I won't say!! Once you have found all three coins and are carrying them you must take them to a small log cabin that lies somewhere in the forest and place them into a chest. Once that is done, your quest is complete. Of course, you must find the key to unlock the door to the cabin! That really is all I can say about the storyline. I wish you successful storing!

Hashbase 128

Get to grips with a fast access database utility

By Steven Burgess

Do you want a program which will give you instant access to your database records? Do you want a program with a comprehensive disk utility built in? Do you want a program which will allow you to customise the program for your printer? (serial device 4 printers only). Do you want a database which, unlike conventional ones, uses fast access and storage routines?

If your answer to any of the above questions is a resounding YES, or even a non-resounding YES, then look elsewhere... No, only kidding.

Yes, believe it or not, this program has all the above and it is written in BASIC!!

It will work on any C128 computer and it is DATASETTE compatible, which, as you already know, is rare.

As mentioned earlier, this program doesn't use a conventional means of data storage. Instead, it uses a storage system called HASH TABLES, hence the title of the program.

HASH TABLES allow you to access your records with remarkable speed, regardless of the number of records entered. The computer does not search through the entire database in order to find one item. Indeed, if you are lucky, your record could be found the very first time the computer looks at your HASHBASE. Something similar to you sticking your hand into a large tub of raffle tickets when only one ticket bears the winning number and, hey presto, when you remove your hand and give in your number you find that you have won some guest shell soap and a royal wedding commemorative thermometer which, in the middle of winter displays the temperature 49 degrees centigrade.

Not only does this program incorporate such a marvellous method of data retrieval placed in the conventional surroundings of a common or garden database, it also provides you with a concise disk utility. Among the many operations this utility will perform are RETRIEVING SCRATCHED FILES, PROTECTING FILES, FORMAT WITH 5 CHARACTER I.D. and much, much more.

So, if you think you could use a program like this, and frankly who couldn't... then read on and be stimulated beyond your wildest dreams, in the nicest possible way.

HOW TO USE IT

The program is operated in very much the same way as an ordinary database. Of course, if you have never used a conventional database then that snippet of information is pretty useless, so let's, shall we, start again.

As the program is menu driven, mainly, I will explain the use of each option in each menu as systematically as possible.

The first, and, when you start, only, option you can choose is 1... CREATE FILE. When this option is selected, you are asked to enter the number of fields you require for your hashbase. As this cannot be changed afterwards you are advised to plan your hashbase as much as possible before wasting time, and your sanity, fiddling about and making mistakes.

After you have chosen the number of records, using the < & ? keys to move the cursor and the RETURN key to choose, the hashbase structure information is displayed. e.g. if you said you wanted 10 fields the following information would be displayed:

HASH BASE STRUCTURE
HASH TABLE: 50 RECORDS
OVERFLOW TABLE: 450 RECORDS

This simply means that whatever data you enter the hash value will always be between, and including, 1 and 50. The data entered will then be stored at this location in the array. If the location is engaged, to use a lavatorial expression, then the data will be stored in the overflow table, at location 51, then 52, 53 etc etc.

Once you have marvelled sufficiently at the information displayed, press any key and then the table will be formatted so that it can accept hash records. The length of time taken for this to complete varies, depending upon how many fields you chose. Then, after all of that, you have to enter the field names pressing return after each one. You are then shown the fields and have to check whether there are any errors, if there are you must correct them and if there aren't you are returned to the main menu.

The next option allows you to add records and is selected by pressing 2. When you do choose this option the field names are displayed and you have to enter the data of your record which corresponds to the file which is being displayed on screen. E.g. if NAME were to be displayed you could type FRED BLOGGS etc.

There are no record numbers, as such. The data is put into the hash base in a location which is calculated by the nature of the data rather than chronological order.

Once you have entered the full record, you are asked whether you wish to add more records. If you do not then you are returned to the main menu.

The next 4 commands are manipulative and accessive. That is they manipulate and access the hashbase records.

The first two, **Amend** and **Delete**, are operated in pretty much the same fashion. When the option, be it **Amend** or **Delete,** is chosen you are asked to enter the keyfield of the record which you wish to manipulate. The key field is the first field. (See diagram 5.)

Provided the record is present, it should be displayed almost immediately. Then you are asked if the record is the one which you wish to manipulate, as more than one record could share the same keyfield. If you press N then another record will appear, if there are any, and the same question will be posed. This will continue until you press Y, you have found the record, or no more records exist.

If you selected DELETE you will be asked if you are sure you wish to delete. If you are the record will be deleted and you will be returned to the main menu. If you aren't you will simply be returned to the main menu.

If, however, you selected AMEND

it is a completely different kettle of ballgames. Instead you are asked if you wish to change the key field. This is essential because if you do, the location in which the record is stored will have to be changed. So the routine, in asking this question, is preparing itself.

Anyway, whichever field, or fields, you wish to change, you are given its current value and asked to enter the new value. When you no longer wish to amend any more fields enter 0 and you will be returned to the main menu.

The next command is VIEW RECORDS. This does not require you to do anything. All it does is run through the entire hashbase displaying the records one by one. You are told that it is a lengthy process and can abort if preferred.

Now here, in the next option, is where hash tables come into their own. The search option, although not terribly sophisticated, allows you, as its name suggests, to search through the hashbase. But before the search can take place you must enter the data which all the records you wish to find contain. If you include the keyfield in the data you enter, then the search will be remarkably fast. If, however, you do not it will take as long as it would to do an ordinary, serial search.

When the search is complete, you are shown all of the records and returned to the main menu.

The final option on the main menu is option 0...**DELETE ENTIRE FILE**. Before taking this drastic action you are asked whether you are sure, twice. But after that...

PRINTER

Next we come to the printer menu.

Selecting option 1, of the printer menu, has exactly the same effect as selecting the search option of the main menu, except, instead of printing to the screen, it prints to paper.

Selecting 2 returns you to the main menu.
3... Toggles between SINGLE SHEET/FANFOLD PAPER
4... Toggles between A4/A5 paper size
5... Toggles between DRAFT/NLQ
6... Toggles between PICA/ELITE/CONDENSED
7... Allows you to enter the codes for all of the following to customise. DRAFT, NLQ, PICA, ELITE & CONDENSED.

The number of records printed per page depends upon the paper size and the number of fields selected. If you choose fanfold paper then the printing is continuous, that is at the end of each page the printer moves to the beginning of the next page and continues. With single sheets, you are asked to insert a new sheet of paper and press any key before printing will resume. When all of the records are printed you will be returned to the print menu.

SAVE/LOAD

The save/load menu, as its very name suggests, allows you to save and load your files. If there is already a file in memory you cannot load. Before you can do either, though, you must enter the filename, by choosing option 4. Selecting option 3 (pay like rental, service like rental...) toggles between devices 1, 8 & 9.

When you do select load, the hashbase is formatted before loading the file. This is done so that all the hashbase can accept hashbase records.
When you save the hashbase is compressed so that all the empty records need not be saved.
Option 4 allows you to enter a filename.
Option 5 returns you to the main menu.

DISK UTILITY

This could be considered to be a completely different program as it does not affect the hashbase program at all.

Quite simply, it is a small program which collects all the fiddly disk commands, and a few extra, together.

Most of the items on the menu are self explanatory so I will only go into detail on the ones which may, to some, be rather frightening at first sight.

1: INITIALISE DISK: This initialises the disk. It causes the disk drive to read the directory information into its memory.

2: VALIDATE DISK: Selecting this option causes the disk drive to 'clean up' the disk by making previously unavailable, yet free, disk space available for use. It can often be quite a lengthy process so be prepared to wait.

3: SCRATCH: As you can probably imagine, this option allows you to delete a file or some files off a disk. The use of the asterix is allowed so enabling you to delete a whole set of files which share a similar name. e.g. entering PRO* will delete all files which have the first three characters in their name as PRO e.g. PROGRAM1, PROCESSOR, PROLOG etc.

4: RETRIEVE SCRATCHED FILE: Allows you to retrieve a previously scratched file from the disk, but you must know the name.

5: DIRECTORY: Displays the directory of the disk in the drive.

6: CHANGE DISK NAME: Allows youto change the disk name and ID of a particular disk. The ID can be 5 characters long.

7: FAST FORMAT: Allows you to format a disk which has already been formatted by a CBM drive at a fast speed.

8: FORMAT DISK: Allows you to format a disk which has not yet been converted to commodore disk format. Instead of the conventional two character disk i.d., you are allowed five.

9: STATUS: Gives the disk error and status line. Use it when the disk light is flashing.

0: PROTECT FILE FROM ERASURE: When selected this option allows you to protect a file of your choosing from accidental erasure.

A: UNPROTECT FILE FROM ERASURE: When selected this option allows you to unprotect a previously protected file.

B: CHANGE FILENAME: Allows you to change the filename of a file of your choosing.

C: RETURNS TO HASHBASE: self explanatory.

DISADVANTAGES

In my view, there is only one real disadvantage. That is the inability to sort the file. But this pales into insignificance when you get used to the new, superfast format.

Minor disadvantages include the compression and formatting routines which are essential for successful execution of program but which lengthen the loading and saving times somewhat.

GETTING IT IN...

...as the actress said to the bishop.

In order to enter the program into your computer simply follow the steps shown below.

1.. Turn you computer on.
2.. Turn your monitor on.
3.. Turn your drive on.
4.. Load the program.
5.. Run it.
6.. Have fun.
7.. Goodbye.

PROGRAM BREAKDOWN

110-400	Display title screen and wait for keypress, then go to menu.
450-490	Routine which prints cursor and awaits a keypress.
540-730	Display menu, take option and go to appropriate routine.
780-1060	Error routine, called whenever an internal error occurs. E.g. No file name.
1110-1310	Get number of fields required.
1320-1600	Display number of fields, number of records and hashtable structure. Also formats table.
1610-1920	Get field names, check for errors, correct errors.
1930-2050	Allows entry of string, length L character to be printed at X,Y.
2060-2150	Calculates the hash equation.
2180-2220	Inserts a record into the hash table.
2230-2410	Allows the user to add records.
2460-2920	Allows user to amend previously entered record.
2940-3230	Allows user to delete previously entered record.
3250-3650	Searches through entire file and prints all the records to the screen.
3690-4060	Searches for a specific record from the data provided by user.
4070-4150	Displays records found in search (3690-4060).
4190-4550	Display print menu and process chosen option.
4570-4870	Print records found from search to printer.
4920-4400	Change printer codes.
5540-5820	Display LOAD/SAVE menu and process chosen option.
5830-60450	Saves file after compacting.
6060-6410	Loads file into computer after formatting hash table.
6460-6600	Delete file and re-run program.
6640-7030	Display disk utility menu and process chosen option.
7040-7080	SCRATCH file.
7090-7190	Retrieve scratched file.
7200-7410	Change disk name.
7420-7720	Fast FORMAT with 5 char I.D.
7730	FORMAT with 5 char I.D.
7740-7870	Protext file from erasure (doesn't need protection, their songs are fab and groovy no less!)
7880-8010	UNPROTECT FILE (Right on! you coooool doods!)
8020-8060	RENAME file
8070-8210	Find location on disk of a specified file.

DIAGRAM 1
MAIN MENU
1... CREATE FILE
2... ADD RECORDS
3... AMEND RECORD
4... DELETE RECORD
5... VIEW RECORDS
6... SEARCH
7... PRINT MENU
8... LOAD/SAVE MENU
9... DISK UTILITY
0... DELETE ENTIRE FILE

DIAGRAM 2
PRINT MENU
1... SEARCH + PRINT
2... RETURN TO MAIN MENU
3... FANFOLD/SINGLE SHEET
4... A4/A5
5... DRAFT/DOUBLE STRIKE or NLQ
6... PICA/ELITE/CONDENSED
7... ENTER PRINTER CODES

DIAGRAM 3
LOAD/SAVE MENU
1... LOAD FILE
2... SAVE FILE
3... DEVICE: 1/8/9
4... FILENAME:filename
5... RETURN TO MAIN MENU

DIAGRAM 4
DISK UTILITY MENU
1: INITIALISE DISK
2: VALIDATE DISK
3: SCRATCH FILE
4: RETRIEVE SCRATCHED FILE
5: DIRECTORY
6: CHANGE DISK NAME
7: FAST FORMAT
8: FORMAT
9: READ ERROR CHANNEL & DISK STATUS
0: PROTECT FILE FROM ERASE
A: UNPROTECT FILE
B: CHANGE FILENAME
C: RETURN TO MAIN MENU

DIAGRAM 5
FIELD EXAMPLE
FIELD NUMBER FIELD NAME
1 AUTHOR KEYFIELD
2 TITLE
3 PUBLISHER
4 TYPE
5 PRICE
6 I.S.B.N.

1 NAME KEYFIELD
2 ADDRESS
3 AGE
The keyfield, in all cases, is field number 1.

Centronics RX Interface

Fed up with incompatibility between hardware? This program could be just what you are looking for

By M.D. Addlesee

There comes a point when the attractions of the latest technology can no longer be resisted. You convince yourself that you really can afford it and on your desk appears a brand new computer ready to respond to your every whim. Now it would be really useful if all your old peripherals, primarily the printer, could be used with the new system. However, due to the lack of standardisation between manufacturers you find the two systems are incompatible.

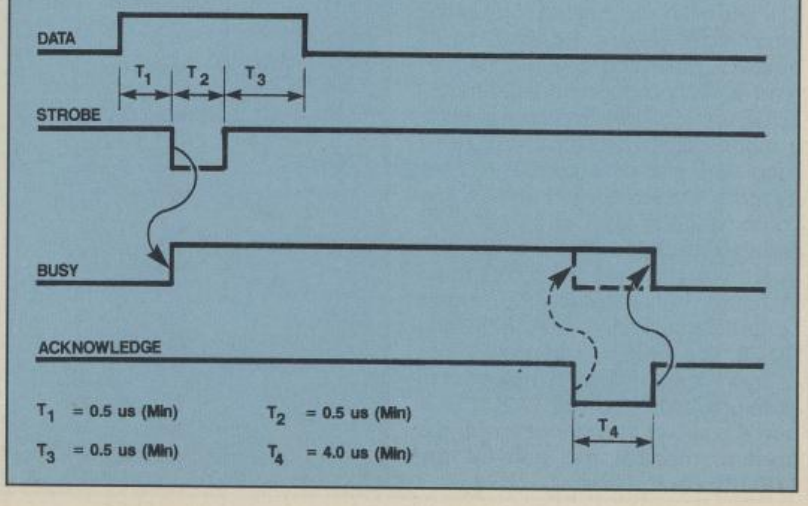
This is the situation I found myself in recently. I had just purchased an Amstrad PC1512 and wanted it to print out to a Commodore DPS1101 daisy wheel printer which had been giving excellent results with my trusty CBM64. The DPS1101 is a re-engineered Juki 6100 which has been provided with a Commodore serial bus interface. This allows the printer to be connected to the CBM64 along with other devices, such as the 1541 disk drive, by a daisy-chain technique that allows simpler wiring between the various units. Easy I thought, all I have to do is write a program on the Commodore 64 to receive data from the PC and redirect it to the printer over the serial bus. Now the standard way of transferring data between two machines is using the ubiquitious RS232 port. I poured a cup of coffee, sat down with the manuals strewn around me and started to read. Twenty minutes later I had discovered three obstacles in the use of this so called standard RS232 interface.

The first problem was that although on a PC compatible you can redirect printer output from the parallel printer output port to the RS232 port using the command MODE LPT1=COM1:, I was already using the PC's RS232 port to communicate with an external modem and I'm too lazy to keep on swapping connecting cable. Secondly, the CBM64 requires the Commodore RS232 interface cartridge or equivalent to be plugged into the user port. This cartridge converts between the user port TTL voltage levels and the plus and minus twelve volt levels defined in the EIA standard for RS232 interfaces. The cheapest cartridge of this type that I have seen sells for £20, not a trivial amount. Finally, the CBM64 kernal cannot communicate over the RS232 channel and the serial bus simultaneously.

So, rather than use RS232, why not use parallel transfer between the two machines? The PC provides a standard Centronics interface at the printer output connector (see Table 1), there is a parallel user port on the CBM that can receive data and parallel data transfer is invariably faster. Let's take a look at the timing diagram for Centronics parallel data transfer to see what is required (see Fig. 1). When the PC sees that the printer is ready to receive data by the **BUSY** line going low, it puts the next character to be transmitted onto the **DATA** lines and signals to the printer that data is present by taking **STROBE** low. The printer must now latch onto this data using the **STROBE** pulse and indicate that it is processing it by sending BUSY high. After some period of time, the printer pulses **ACKNOWLEDGE** low to tell the PC that it has received and processed the data i.e. it has either printed it or stored it in an internal buffer to await printing. The low pulse on ACKNOWLEDGE is coincident with BUSY going low in readiness for the next data transfer.

Now a problem emerges because the 6526 Complex Interface Adapter (CIA) chip used to implement the user port needs to have its port B peripheral data register (PRB) read as soon

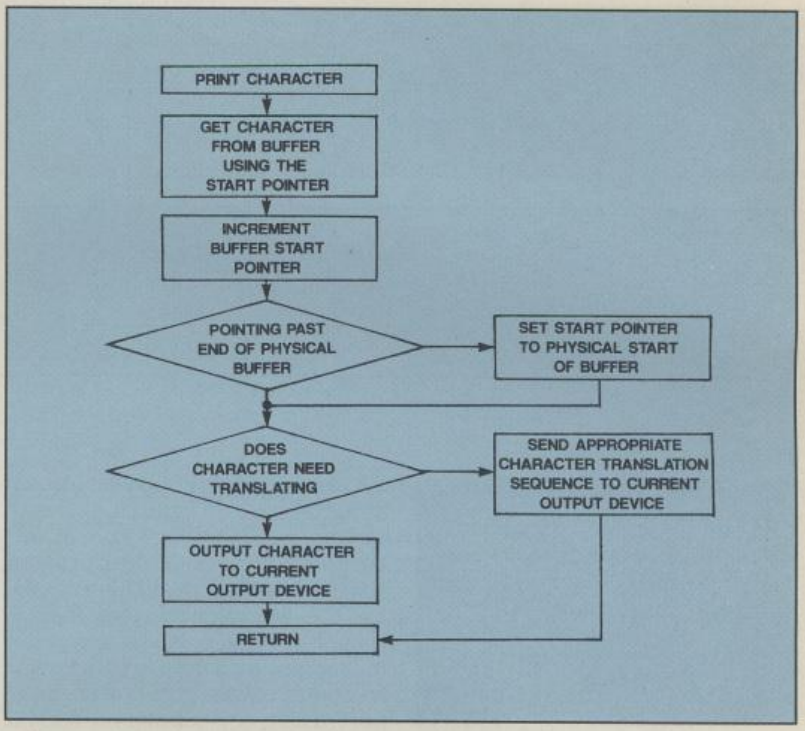
FIG. 1 CENTRONICS TIMING CHART

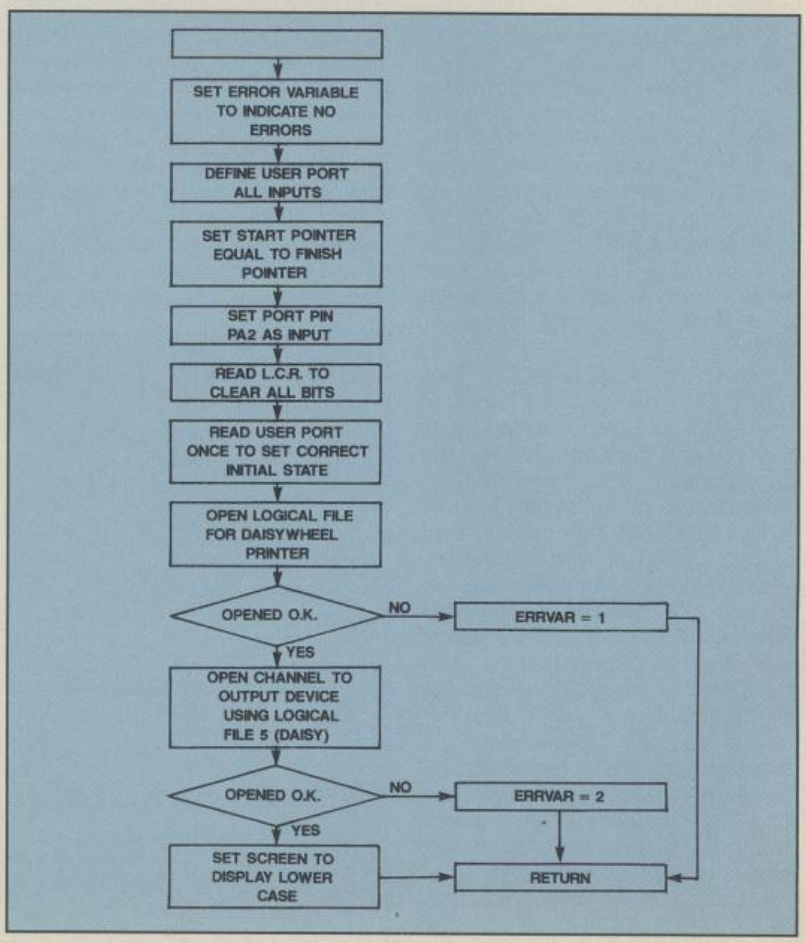


as the data is presented on the eight data lines. The fastest that this can occur is under interrupt with STROBE connected to the FLAG2 line on the user port, as the source of the interrupts. The 6510 microprocessor in the CBM64 will take several microseconds to respond to the interrupt and read CIA PRB by which time the data could be long gone (minimum time for which Centronics data can be present is 1.5 microseconds). Also bearing in mind the precise timing requirements for the various transitions on lines STOBE, BUSY and ACKNOWLEDGE which cannot be met under software control, we conclude that the CBM64 user port interface as it stands is incapable of meeting the requirements for Centronics parallel data reception. All is not lost though. With just four TTL integrated circuits we can add all the extra functions the Commodore 64 needs to allow its user port to receive data to the Centronics standard.

Figure 2 shows the circuit diagram for turning the user port into a Centronics compatible data receiver. This entire circuit can be squeezed onto a printed circuit board a couple of inches square and plugged into the user port. To keep the cost down the 4*SPDT DIP switch can be replaced by simple wire links and the 74LS574 by a 74LS374 BUT NOTE THAT THE 74LS374 HAS A DIFFERENT PINOUT. All the parts should be readily available from good mail order electronic component suppliers.

Operation of the circuit is quite straight forward. Logic gates U3A AND U3B form an RS flip-flop that ensures correct transitions on the various control lines including the CLK line of U1. U1 takes a copy of the incoming data and holds it on its outputs until the processor is ready to read it. To produce an ACKNOWLEDGE signal of the correct pulse width a monostable multivibrator, U2B, is used with the time constant determined by the values of R1 and C1. The combination of U3C and switch one allows us to choose between the two options for the timing of BUSY shown in figure 1. U4 buffers the output control signals BUSY and **ACKNOWLEDGE**. The three remaining switches of figure 2 are shown in their normal positions. Note that bit PA2 of user port A is set up as an input line and is used to monitor the state of the **STROBE** control line.





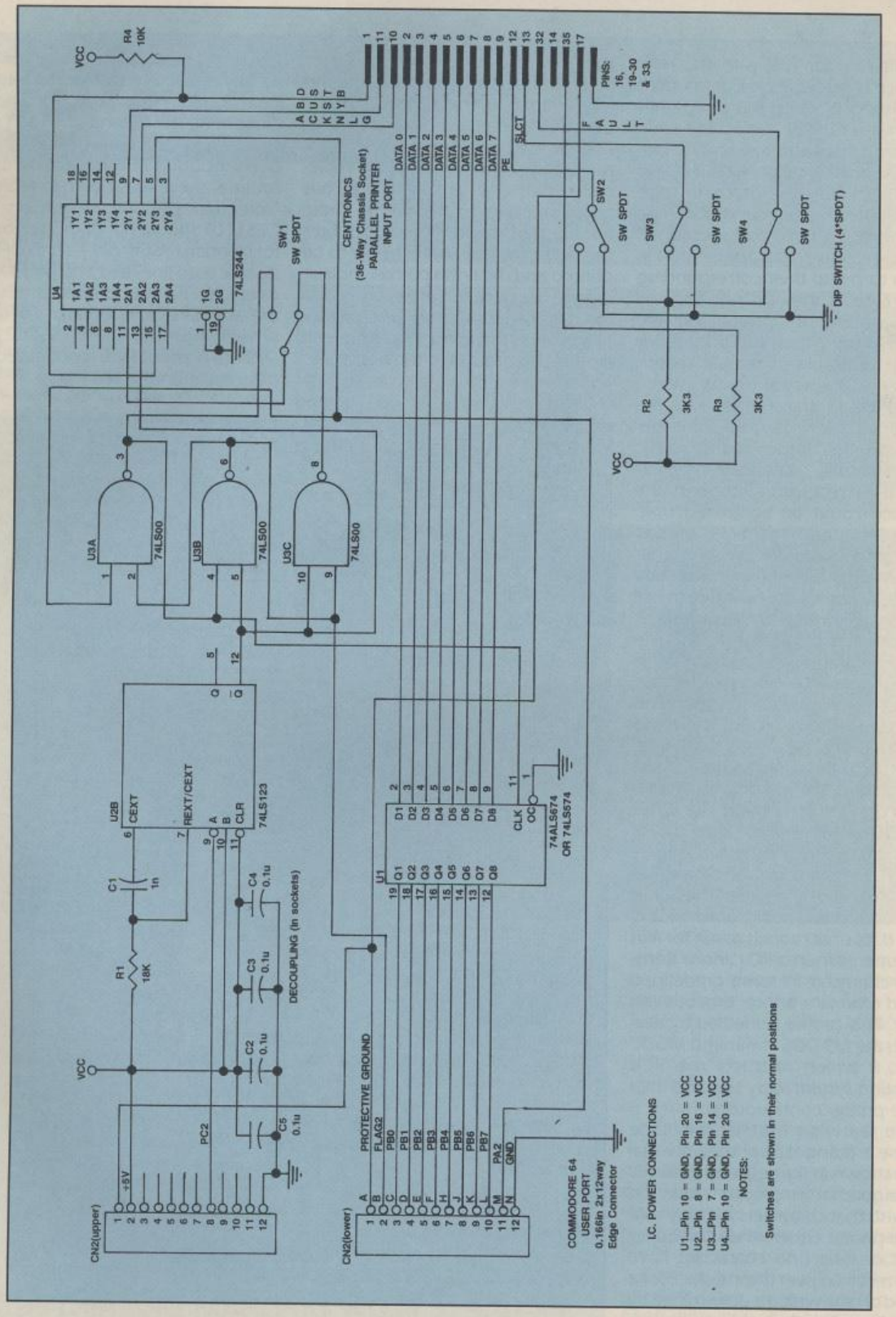


FIGURE 2. INTERFACE CIRCUIT DIAGRAM

Software for use with the new interface is given in the Commodore Basic listing 1. When this program is run on the CBM64 a machine code program is inserted into memory starting at decimal location 49152 (C000 in hex). To start the machine code routine enter the command SYS49152 and then hit return. You should see any upper case characters on the screen turn into their corresponding lower case. if they do not and the ready statement appears on the CBM64 display then the program is unable to establish communications with the daisy wheel printer. When data is being transferred over the Centronics interface from the other computer the border colour of the CBM64 display changes once for each byte received. Although the interface could be controlled from basic, data transfer is much quicker when machine code is used. Another advantage of using the machine code routine is that we are now free to use the memory normally reserved for basic programs as a large 38912 byte circular buffer. Rates of transfer as high as four thousand characters per second have been achieved. This means that when printing files whose combined size is less than that of the buffer the data is sent to the CBM64 in a matter of seconds leaving the PC to get on with something else while the CBM64 handles the slow job of sending characters to the daisy wheel printer. If the amount of data to be printed exceeds the size of the buffer the PC is prevented from sending more data until room is made for it in the buffer. When a PC cannot transmit a character in some predefined period of time a printer timeout can occur. This can be remedied by executing the MS-DOS command MODE LPT1:,,P which instructs the PC's operating system to try sending data to the printer continuously.

To see what the machine code routine is doing look at the flow diagram shown in figure 3 and listing 2. The algorithm employed works on the basis that characters are only sent to the printer either when the buffer is full or when no characters have been received over the interface for a period of time which is determined by the value of the variable 'tries'. It is important to select this value carefully to ensure maximum data transfer rate. With the code as it stands (tries = 255) the period between reception of continuous data bytes should be no longer than 3.9ms. Polling of the Centronics receiver interface was chosen in preference to using interrupts because of simplicity of implementation. The circular buffer has two pointers associated with it. These are the start pointer which indicates the next character in the buffer to be printed and the finish pointer indicating the next location at which incoming data can be stored. When either pointer gets to the end of the buffer it is wrapped around to the beginning again hence the term circular. As data is pulled out of the buffer it is checked for any character translations that may be required. For maximum compatibility between PC and DPS1101, six special character translations have been included in the machine code routine, these are shown in table 2. Note that DIP switch one on the DPS1101 should have pin 3 set to ON for standard ASCII, pin 1 set to OFF to disable auto line feed and pins 4, 5 & 6 set to OFF if your printer came with the standard JUKI courier 10 daisy wheel number 176-OAO. The DPS1101 should be configured as device five on the serial bus.

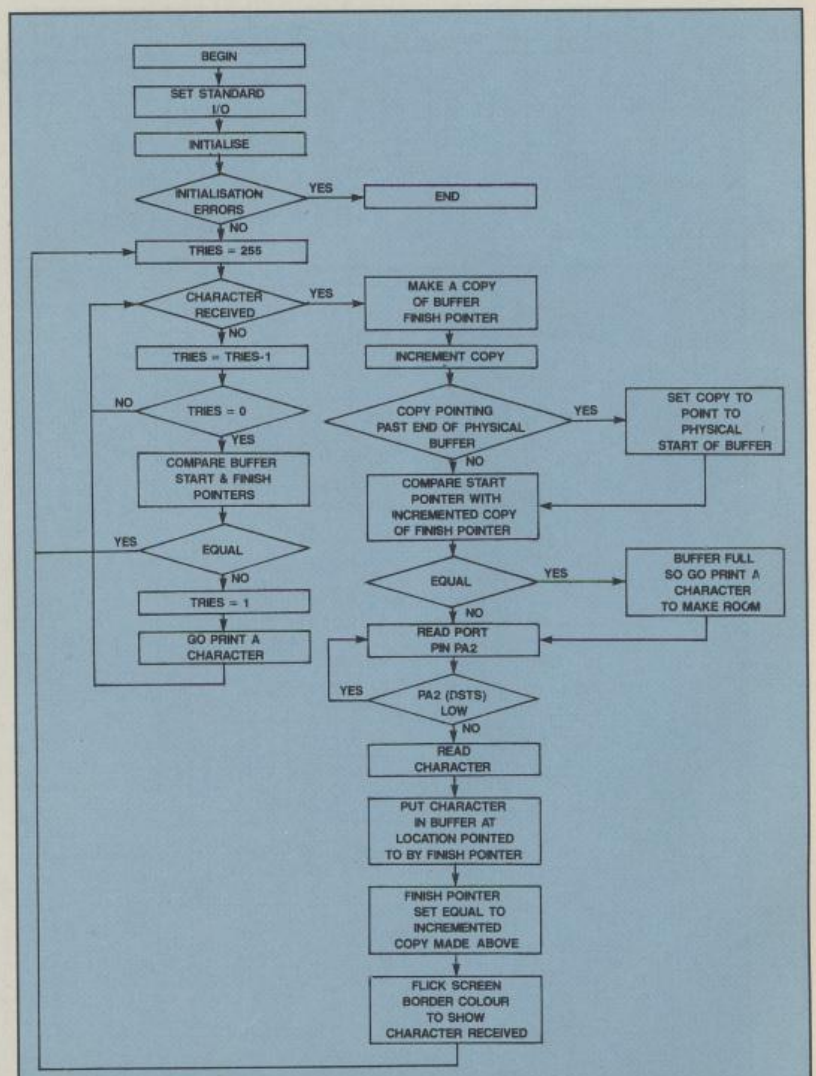


FIGURE 3. FLOW DIAGRAM OF PRINTER BUFFER ROUTINE

Revasm C64/C128

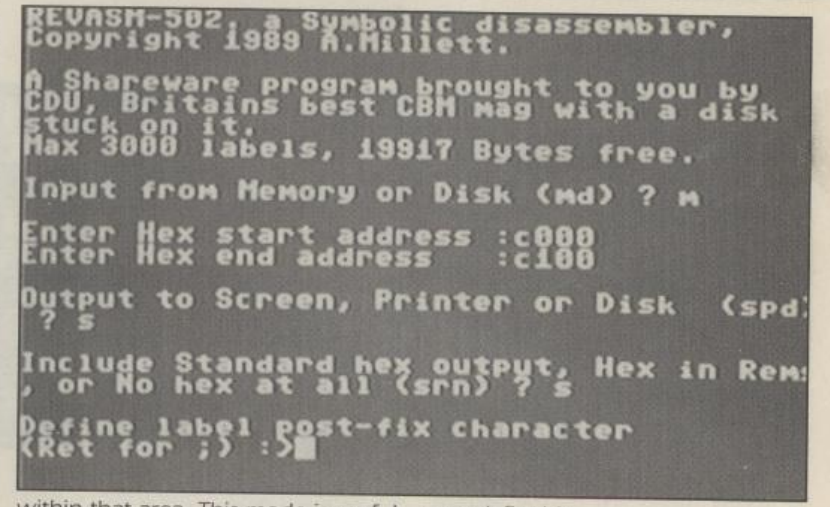
Two more unassemblers see the light of day, one for the C64 and the other for the C128

By Adrian Millett

Sadly I looked at the inert Commodore PC1, and contemplated the ordeal of packing it up and sending it back under warranty. Still, while it was being repaired, there are some CBM-64 projects that could profitably be finished off. One old favourite of mine, of ancient history, needed "Resourcing" because it had been subject to the bad habit of direct machine-code patching (on an 8K PET!). To turn machine code back into assembler source with proper labels requires a labelling disassembler. Unfortunately the only symbolic disassembler I had in my library has serious limitations – the chief one being that it could only disassemble from memory, so it can't cope with code that sits in the same location as itself! So, working from a normal vanilla disassembler I coded many moons ago, I have written a new one toy specification. The result? REVASM 64/128.

REVASM will accept machine code, from a disk file or memory, and turn it into a screen/printer listing or a SEQ disk file suitable for most assemblers. It will also optionally accept a list of user labels and locations (from a disk file or direct from the keyboard) and intelligently incorporate them into the resulting code. For example:- the user can specify location $FFD2 as label WRITECHAR, and REVBASM will generate the code JSR WRITECHAR instead of JSR $FFD2. And there's more – you can specify areas of the code as being CODE, DATA or VOID. If an area is CODE, normal machine code source is generated. If an area is specified as DATA, REVASM will generate hex bytes in the form:-

.BYTE $xx,$xx,$xx,$xx... Any labels will be interleaved in the normal way. If you specify an area as being VOID, REVASM will generate nothing! Well actually, almost nothing – it merely generates:- * = * + $xxxx statements, again interleaves by any labels within that area. This mode is useful when there is so much data that it is better saved off as a separate binary file.



When you come to use REVASM, you will notice that labels appear on a line on their own, rather than the more traditional format of having source mnemonics following on the same line. I have done it this way because:-

1) The vertical gaps/indentations in the code make reloop + subroutine start points easier to use.

2) Since labels can be any length, there would be occasions when mnemonics would be pushed far over to the right, resulting in a mess.

3) It's easier to code REVASM to do it this way!

The post-fix to a label, which can vary from assembler to assembler, is user definable.

REVASM is mostly self-explanatory in use, since program prompts provide most of the info you need. Generally speaking, hitting RUN/STOP will abort REVASM and get you back to BASIC. When you run it, a copyright message is displayed followed by a question:-

"Input from memory or disk (md)?"

Simply select the appropriate option by typing 'm' or 'd'. If you select disk, you must have the executable PRG file on the current disk in drive 8. If you select memory, you will then be asked for a start and end address. Type these in with a <RETURN> after each 4-digit hex address, ie. 0b00<RETURN> 0b80<RETURN>. If you disassemble from memory, it is important to remember that REVASM sits in memory itself! REVASM resides

between hex locations $1c00 and $ff00 in bank 0. For this reason you are generally better off disassembling from disk. You will now be asked to specify:-

"Output to Screen, Printer or Disk (spd)?"

If you select 'd' for disk, you will also be asked to supply a filename, which will later become a SEQ file on the disk. This file would be the one you could re-assemble later. Now you are asked:- "Include standard hex output, Hex in rems, or No hex at all (srn)?"

If you select 's', the computer will generate addresses and hex at the left hand side of the output, in the traditional disassembler manner. This format is no good if you wish to re-assemble the code at a later stage.

If you select 'r', the computer will generate hex in rem statements at the end of each line, useful for later debugging. However this does generate much longer SEQ output files.

If you select 'n', the computer will generate no hex at all. This is generally the best, as it keeps the size of any SEQ output files to a minimum. The computer's next question is:-

"Define label post-fix character (Ret for ;) ?"

This option allows you to vary the character(s) following labels for compatibility with different assemblers. Most assemblers, however, will allow you to use a semicolon, so in most cases you will just hit <RETURN>. You are now asked:-

"Do you want to enter user labels from Keyboard, Disk or Not at all (kbn)?"

If you select 'n', the computer will generate its own labels in the source output. These are made by simply taking the hex address value of the label and attaching the letter 'z' if it is a zero page location, or the letter 'v' if it is an absolute address:- ie. z8f, v033c etc.

If you select 'k', you will be allowed to type in your own label names. The computer asks you:- "Enter Label (Ret to end) :" When you have entered the user-label name, followed by <RETURN>, you are asked:- "Enter Hex value :" Now you enter the value for that label. Repeat this process until you have entered all your labels. When finished, simply type <RETURN> on a blank input. For example, you could type the following:-

time<RETURN> a0<RETURN>
writechar<RETURN> ffd2<RETURN>
readchar<RETURN> ffe4<RETURN>
<RETURN>

If you selected 'd' for disk on the earlier option, you will be asked to enter a filename. When you do so, the computer will read your user defined labels off a SEQ file on disk. Each label on this file should be followed by a comma and then the label value. The list must be terminated by a comma followed by <RETURN>, ie:- LABEL,HEX<RETURN> LABEL,HEX<RETURN> ... ,<RETURN> The file needs to be prepared before you start using REVASM. This may be done using a text editor, like the ones supplied with many assemblers, or it may be done using a short basic program like this:-

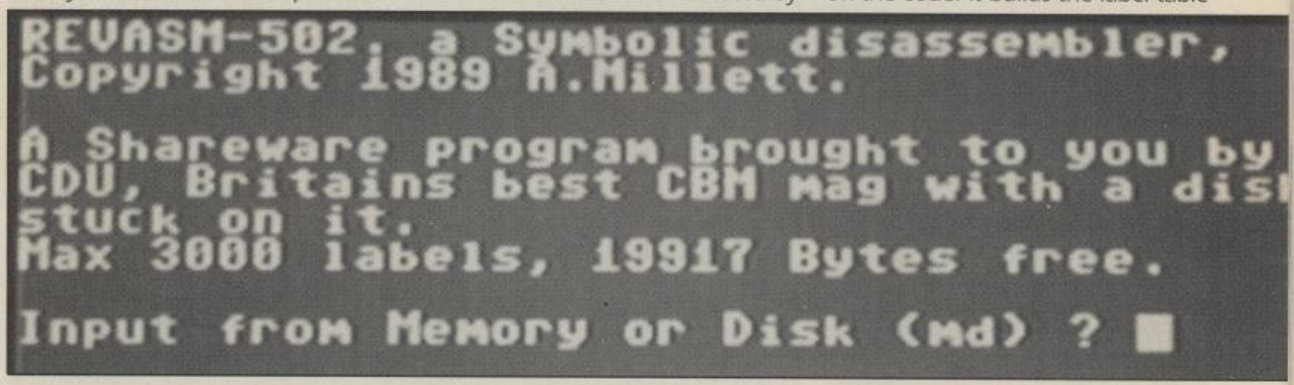
```
    100   restore
    110   open 8,8,8,"prog.lab,s,w"
    120   read lab$
    130   print#8,lab$
140   if lab$<>"," then 120
150   close 8
160   end
499   :
500   data "time,a0"
510   data "writechar,ffd2"
    520   data "readchar,ffe4"
    990   data ","
```

As I mentioned earlier, it is possible to specify areas of your code as being CODE, DATA or VOID. This means you can make the computer generate normal disassembly for CODE areas, data in the form of ".byte" instructions for DATA areas, or nothing at all for large areas of data that you wish to deal with in a different way (VOID areas). To specify an area, you simply attach &code, &data or &void to a label in your user-label list. This label should contain the address of the start of the area. For example, you might add the following lines to the above basic program:-

```
530   data "progstart&code,0b00"
540   data "lookuptables&data,0b80"
550   data "moreprog&code,0ba0"
560   data "graphics&data,0bc0"
```

Having selected all the necessary options, the computer will whiz off and generate the source doe. You can hit the left arrow to pause at any stage, and you may use RUN/STOP to abort. The computer does 2 passes on the code. It builds the label table on the first pass, and generates the source code on the second. If it comes across any illegal opcodes, it will generate an appropriate .BYTE instruction. Watch out for labels that refer to locations within an instruction. These will have a rem statement saying "Dangerous label!" attached to them. This indicates a potential problem, especially if you want to re-compile the source at a different location. Generally speaking, you may have to do a number of test-runs before you work out where all your CODE, DATA and VOID areas are, and the values of all your user-defined labels. An extra point for the CBM-128 version of the program:- To re-run the program after exiting to basic, type:- BANK 0:SYS 56139 <RETURN>.

Well, that's it. I hope you find REVASM as useful as I did!



Speedy Unassembler

Especially for those of you that use the Speedy Assembler from Your Commodore, here is the unassembler to compliment it.

By Mike Gregory

Many readers will be familiar with an 'assembler'. This is a utility programme which converts a file, commonly called 'source code', into machine code which is then directly executable on the Commodore 64. The source file is a special form of pseudo-code which allows us poor humans to understand machine code operations. It makes use of labels to designate both branch points and variables, and mnemonic operators to represent the opcode numbers which comprise the instruction set understood by the Commodore 64.

An 'unassembler' does the reverse. It converts a machine code file into labelled source code.

Why is it useful?

Most programmers, from time to time, develop a need to examine machine code for which there is no source code listing. Often we wish to understand the programme's operation or to follow its structure and flow. Perhaps we would like to relocate the machine code so that it will run at another place within the Commodore 64's memory, or perhaps we would like to modify some sections of the code so as to fit our own requirements. Maybe we are simply trying to learn how to improve our own programming.

In all these cases, a utility which can read the machine code and convert it into a source file fit for human consumption, and which can be re-assembled by our assembler, becomes essential.

Once the source file is available, it can be studied or modified with ease. For example, to change a programme's location in memory, we often only need to alter the 'ORG' directive in the source file and re-assemble. Of course, to modify a programme successfully, it is essential to understand how it works, and again the source code is invaluable.

How does it work?

The Speedy Unassembler performs its tasks in a series of passes. Initially, in Pass 1, it prompts for the name of the machine code file, locates it on the disk and reads through it to find the start and finish loading addresses.

In Pass 2 it obtains a name for the source code file. The name of the machine code file can be used as a default option if required. This pass also asks for the unassembly start and finish addresses, together with any regions of the machine code known to represent non-code areas such as tables of characters, data bytes, addresses, etc. These latter addresses are non-essential but reduce the occurence of spurious coding within the source file. Supplying the table information generally assists in producing a more intelligible listing.

Pass 3 reads a section of the machine code into RAM and at the same time collects all the labels involved in the batch. The length of the batch is set to about seven disk blocks, or the whole file if it is shorter.

During Pass 4, the labels are sorted into ascending order and any multiple occurences of the same label are removed. Don't blink or you'll miss it!

Pass 5 compares the labels with the RAM based code and separates out those labels which relate to addresses within the code. These are the 'internal labels'.

The remaining labels, known as 'external labels', are written into the source file as an 'equates' section in Pass 6.

The real work is done in Pass 7. The internal labels together with the mnemonic pseudo-code and any byte tables are added to the source file until the RAM based code is completed.

If more code is to be unassembled, that is if the original machine code file is longer than seven disk blocks, Passes 3 through 7 are repeated until the whole file has been unassembled.

For information, the processing speed for actual unassembly (Passes 3-7) is about 40 secs for an average batch of machine code. During this time seven blocks of machine code

will be read from disk, converted to something like 40 blocks of source code and this source code will be written back to disk. The length of the source code will vary depending upon such things as number of labels, presence of byte tables, etc, but it can be expected to be from five to seven times the length of the original machine code.

How is the Speedy Unassembler used?

The easiest way to learn how the Unassembler is used is by example. Listing 1 is a short programme in a form suitable for assembly by the Speedy Assembler.

Enter the programme exactly as and correct it. Just to make sure, try running the programme with,

: RUN

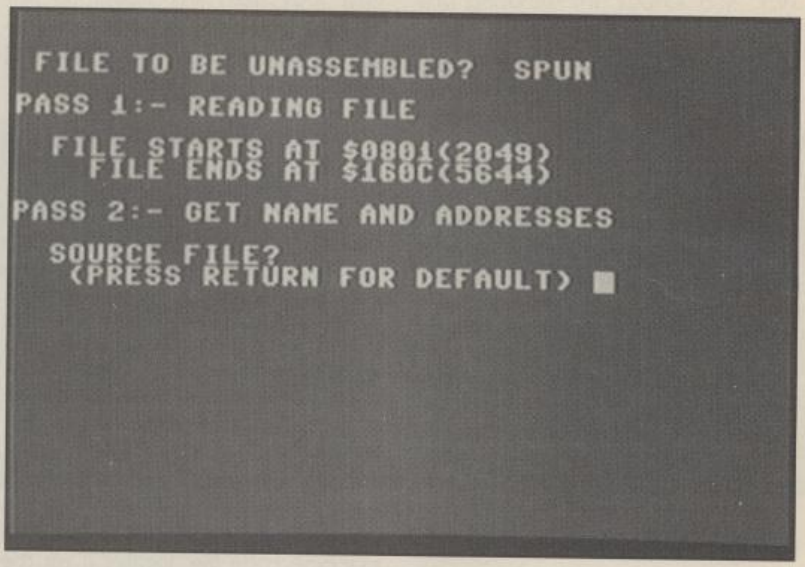
If everything is ok you will see a classic message flash briefly on the screen. Try running it again if it was too quick. The assembled code can be examined using,

M $0801-$0840

The '60' at $0834 corresponds to the RTS at the end of the programme.

LISTING 1

```
10    ; EQUATES
20    CLEAR EQU $93
30    STROUT EQU $AB1E
40    CHROUT EQU $FFD2
50    ;
60    ; LOCATE CODE AT START
       OF BASIC
70    ORG $0801
80    ;
90    ;SET UP BASIC PROGRAMME
100   ; '10 SYS 2061'
110   WOR LINK
120   BYT 10,0
130   BYT $9E, "2061",0
140    LINK BYT 0,0
150   ;
160   ; CLEAR SCREEN AND PRINT
       MESSAGE
170   LDA #<TEXT ; LOW BYTE
180   LDY #>TEXT ; HIGH BYTE
190   JSR STROUT
200   ;
210   ; WAIT FOR A WHILE
220   JSR PAUSE
230   ;
240   ; CLEAR SCREEN AND FIN-
       ISH
250   LDA #CLEAR
260   JMP CHROUT
270   ;
280   TEXT BYT $93, "HELLO
       WORLD",0
290   ;
300   PAUSE LDY #0
310   LDX #0
320   P1 INY
330   NOP
340   BNE P1
350   INX
360   BNE P1
370   RTS
```



Notice how the message stands out. When the assembly has been done successfully, save the machine code as, (note that the symbol '<-' is used to represent the left-arrow single key)

<-CODE, $080-1-$0834

Exit Speedy Assembler using 'B' and crank up (load and run) Speedy Unassembler. Enter 'CODE' at the first prompt and when the addresses are given select the default options (press RETURN) for the source file name, for the unassembly start address, for the table start address and for the unassembly finish address.

When the unassembly finishes, re-run it again selecting 'CODE' but this time enter 'SOURCE2' for the source file name. Select the following addresses when prompted,

When it finishes, re-enter Speedy Assembler using 'SYS 20000' and load

Start of unassembly	[RETURN]
Start of table	$0801
End of table	$080C
Start of table	$081C
End of Table	$0828
Start of table	[RETURN]
End of unassembly	[RETURN]

the first source code file,

CODE.A

Notice that the Unassembler has appended '.A' to the default file name. If more than one batch had been processed, the series '.B', '.C', etc would have been appended. Also since DOS limits file names to 16 characters, the source file name length is reduced to 14. If you enter more than 14 characters, the name will truncate to 14 however it will still be unique because of the dot-letter suffix.

If everything has been entered correctly, listing the source file will show the pseudo-code given in Listing 2.

Examination of Listing 2 will re-

LISTING 2

```
10  KK00      EQU $00
20  KK44      EQU $44
30  KK4C      EQU $4C
40  KK083F    EQU$083F
50  KK204F    EQU $204F
60  KKAB1E    EQU $AB1E
70  KKFFD2    EQU $FFD2
80  ;
90            ORG $08701
100 ;
110           BYT11
120           PHP
130           ASL
140           BRK
150           BYT 158
160           BYT 50
170           BMI KK083F
180           AND (KK00),Y
190           BRK
200           BRK
210           LDA #$1C
220           LDY #$08
230           JSR KKAB1E
240           JSR KKO829
250           LDA#$93
260           JMP KKFFD2
270           BYT 147
280           PHA
290           EOR KK4C
300           JMP KK204F
310           BYT 87
320           BYT 79
330           BYT 82
340           JMP KK44
350 KKO829    LDY #$00
360           LDX #$00
370 KK082D    INY
380           NOP
390           BNE KK082D
400           INX
410           BNE KK082D
420           RTS
```

LISTING 3

```
10  KK00      EQU $00
20  KKAB1E    EQU $AB1E
30  KKFFD2    EQU KKFFD2
40  ;
50  ORG$0801
60  ;
70            BYT 11
80            BYT 8, 10, 0,
90            158, 50
100           BYT 48,54,49,0,0
110           BYT 0
120           LDA #$1C
130           LDY #$08
140           JSR KKAB1E
150           JSR KKO829
160           LDA #$93
F10           JMP KKFFD2
F.NO
170           BYT 147, 72, 69,
              76, 76
180           BYT 79, 32, 87,
79,           82
190           BYT 76,68,0
200 KK0829    LDY #$00
210           LDX #$00
220 KK082D    INY
230           NOP
240           BNE KKO82D
270           RTS
```

veal how labels are derived from the machine code. Wherever a hexadecimal address, whether it be in zero-page or elsewhere, is found it is promoted to a label by prefixing the address with 'KK'.

Consideration of the pseudo-code, indicates odd-looking machine code, and consequently the presence of byte tables, in lines 110-200 and 270-340. It can be seen that these areas have thrown up a few unnecessary labels such as KK44 and KK083F. It should be noted though that re-assembly of this listing will produce the original programme. Try it using the [F7] key followed by :RUN.

If now the alternative unassembly is loaded, SOURCE2.A, it should give Listing 3.

This time the listing is far more intelligible. No labels have been derived from the byte table areas. Also the facility to have a series of byte values after each 'BYT' directive has been utilised. Again however it should be noted that re-assembly will still produce the original machine code. Try assembling and running again. The benefit gained by indicating tables comes only in avoiding spurious code sections and in enhancing the readability of the programme structure.

There are a number of ways in which table regions can be detected. For a short piece of machine code such as in the example, a first unassembly run without tables can be made and the listing then inspected for the 'BYT' directive using the Speedy Assembler search command. by making a note of the approximate addresses, the unassembly can be re-run as required with table values supplied until the listing is acceptable.

For longer code however, this becomes somewhat tedious and a better way, which I recommend, is to use a machine code monitor such as MICKMON to do a preliminary scan of the code using the disassembly feature to look for obvious non-code sections. Make a note of the addresses involved and use these as the table values in the unassembly. It is not important that these be 100% accurate. Re-assembly will still produce the proper code. Also in most cases manual editing of the source file is carried out so as to provide more meaningful names for the labels produced by the Unassembler and also to insert comment lines as the programme structure and flow is understood.

Banks and Memory

A simple program that calculates all those awkward pokes for you automatically when reconfiguring the graphics banks and screen memory

By Jason Finch

The list of topics that are not mentioned in the comic that is provided with each Commodore 64, otherwise known as the User's Manual, is almost endless – machine language, raster interrupts, bitmap mode and even multicolour mode and user defined graphics. And why not admit it – graphics certainly make a difference to the appeal of a game. One of the more complex aspects of graphics and memory control is bank switching and memory area selection – again, not one reference to be found in the User's Manual.

Anyway, enough of pulling the manual apart (although that's probably the best thing anyone could do with it!). In this article I shall explain the fundamental requirements and methods that should be used to provide flexible control of the memory and character definition locations including a more detailed explanation of the important locations and how to position and display bit-mapped screens in different areas, as well as providing a formula for calculating the necessary values to control sprite pointers in different banks. This will lay the path to putting sprite data out of the way of BASIC programs quickly and simply without having to re-define the entire character set. I have also designed a program that will tell you exactly what should be POKEd to where to provide you with the necessary configurations, should you find it all a bit too technical. One further point – it is essential that you understand the terms 'bit' and 'byte' before continuing to read. It is also an advantage if you have a knowledge of binary.

First, though, I will clarify the term "bank switching" and why it is sometimes necessary. But before that, some general information. One chip inside the Commodore 64 is responsible for all of the graphics facilities on the computer – ranging from the simple 40 column by 25 row text screen to more complex multicolour bit-mapped screens and raster-scan graphics. It is the 6567 Video Interface Chip, referred to most commonly as the VIC-II chip (or simply the VIC chip). On the standard screen there are 1000 locations beginning at 1024 in memory. Each of these may contain a number in the range 0 to 255 which determines which character is visible at that point and a corresponding colour byte, the map for which begins at 55296. Unlike the screen memory area this cannot be moved.

Although the Commodore 64 has, as you might imagine, 64K of memory, the VIC chip can only access and manipulate 16K at a time. Each of these blocks of 16K is called a bank and there are four of these on the 64. There are two memory locations that allow you to select which of these four banks the computer should use for its graphical information. These are located in the 6526 Complex Interface Adaptor (CIA) #2. Bits 0 and 1 of location 56576 ($DD00) are used to control the bank selection and must be set to outputs by setting both bits 0 and 1 of location 56578 ($DD02). The method for this is described a little later as for the moment I would like to concentrate on the theoretical side of graphics interfacing. When the computer is switched on, both sets of two bits are set to one, thereby selecting bank zero (0-16383 of memory). That is the reason why sprite data and user defined characters must usually lie between these locations.

The programmer can choose where the user defined characters are placed as well as the sprite data. The screen memory area can also be changed from 1024 according to your wishes, and with it change the sprite pointers. The first always lies exactly 1016 bytes further on than the start of screen memory. Therefore, with the screen at 1024 ($0400), the sprite pointers are at 2040-2047. The formula for calculating the start of the sprite pointers is:

1016+(PEEK (53272) AND240) *64 + (3-(PEEK (56576) AND3)) *16384.

I appreciate this looks a bit daunting but by the end of the article you should be able to refer back to it and understand what it does. Try PRINTing the result on power-up of the computer and you should get the answer 2040. With the changing of screen memory, sprite pointers and so on, the range over which sprite definitions can be displayed also changes.therefore if bank one was selected which consists of memory locations 16384 to 32767, with the necessary sprite pointer set to zero, the computer would get its information for the sprite block from location 16384, and if the pointer was a one, from location 16384=64=16448.The byte in memory that controls the positioning of screen memory and character definitions is found, as you might expect, in the VIC chip. It is register 24, location 53272 ($D018) in memory. The value stored here consists of 8 bits, four of which determine the location of screen memory, three that indicate the start of the character set, with bit 0 having no significance. I shall refer to the high nybble (bits 4 to 7) and the low nybble (bits 0 to 3) throughout the rest of the article. To recap using this terminology, the high nybble controls screen memory and the low nybble, character information.

There are 16 possible positions for screen memory in each bank and each screen occupies 1K, fitting rather well into the 16K available in each bank. The values held in the bits of 53272 represent the offset from the start of the bank to the screen memory. To find out where screen memory is, get the four bits in the high nybble to give a value of 0 to 15: (PEEK(53272) AND240/16. This should then be multiplied by 1024 to find the basic start of screen memory. Then, finally, add the start address of the bank (this is simply the bank number multiplied by 16384). As an example, when you power-up the computer, location 53272 holds the value of 21. The value of the upper nybble is then 16, the related bits being 0001 (binary). This, if you use the above formula, gives 1x1024+0=1024, the usual start of screen memory.

The lower nybble has three significant bits that indicate the location of character definitions or the start of the bit-mapped display. There are 8 possible locations and each character set occupies 256x8=2048 bytes. Again this fits nicely into the 16K available. you will probably be aware that with user defined graphics at 12288 and the screen at 1024 you should POKE53272,28. The lower four bits are then 1100. This value is 12 in decimal, and should be halved and multiplied by 2048 to find the start of the character set 6x2048-12288. This is then the principle of the workings of location 53272.

Let's step through the process for putting the screen at 32768 (bank two) with the character set at 36864. The offsets are then 0 and 4096 respectively. This means the high nybble's bits should be 0000 and bits 1 to 3 should resemble the value 4096/2048=2. these will therefore be 010. The other bit does not matter and so the complete byte is 00000100 which is 4. Then if bank two had been selected the screen and character data would be positioned as above when location 53272 contained a value of 4. The same process would be used if you wished to display a bit-mapped screen. The data to produce the necessary colours is usually POKEd to the screen memory area as well as to 55296 if you are using multicolour graphics. With the above example, the screen memory would be at 32768 still and the start of the actual bit-map data, with bit 5 of location 53265 set for bit-map mode, would be 32768. What?? No, it's not one of those horrendous examples of typing errors that send everyone silly because they can't figure out why what is printed is correct, when in fact it's not! Read on...

The Commodore 64's bit-map display is 320 pixels by 200 pixels, a total of 64000 bits. This represents 8000 bytes, near enough 8K. In each bank you will therefore only have room for two bit-mapped screens although, theoretically there are 8 positions. These are with offsets of 0 and 8192 from the start of the bank. If you are planning on using any other location (by setting the appropriate bits of location 53272) then terminate your plans right now! Whatever, you make bits 1 to 3 you will find that the computer will automatically start the bitmap at the next lowest address possible (either the zero or 8192 offset). For example, setting location 53272 to 28 would leave screen memory at 1024 but the three bits concerned with character information and bit-map data would be 110. This creates a theoretical offset of 12288 for the start of the bit-map. The whole lot would then have to take up locations 12288-20479 which is just not possible because it crosses the bank boundary of 16384. So the computer will automatically set the start of the bit-map at 8192. With bit-mapped screens you can think of bit three of location 53272 setting whether the map begins with a zero offset (the bit is zero) or with an offset of 8192 (set the bit to one). The rest of the low nybble makes no difference with bit-map displays.

I chose the example with the screen at 32768 and character definitions at 36864 as, providing you were not in bit-map mode, you would have the standard character set duplicated. This is because in both banks 0 and 2 there is a ROM image of the character sets. Usually the character set occupies locations 53248-57343 ($D000-$DFFF) in parallel with the VIC chip graphics, colour memory and so on, in much the same way that there is RAM "behind" the ROM at $A000 and $E000. However, there is a way of producing user available RAM at $D000 to $DFFF as well – but that's another story.

An image of the ROM character patterns appears to the system at 4096-8191 ($1000-$1FFF) and at the corresponding locations in bank 2, 36864-40959 ($90-00-$9FFF). This can therefore be utilised by the programmer in bank 2. Sprite data and user defined characters can then be stored in bank two, thereby occupying none of the memory that a relatively long BASIC program needs. This imaging is only available in banks 0 and 2, and only applies to character data as seen by the VIC chip. Just like any other RAM it can be used by programs and will not disrupt the character set. However, your own character sets cannot be defined in the aforementioned regions. If you have to place your character set at the position where the images occur (for whatever reason), simply change to bank 1 or bank 3 and use the corresponding addresses.

An illustration of the imaging in action is simple to obtain. Read the contents of location 53272 on power-up. It should be 21 or, if you have changed to lower case, it will be 23. The low nybble is then 5 or 7 with the corresponding three bits (bits 1 to 3) being 010 and 011 respectively. Calculate the theoretical address of the start of the character set using the previous formula (2x2048=4096, 3x2048=1644) and you will find that the answers are the locations of the ROM images for upper (4096) and lower (6144) case letters. To find these for bank two simply add 32768. Remember, you cannot locate your user defined characters in the area occupied by the ROM images.

Now back to how to change the banks and another example. First you should ensure that bits 0 and 1 are set to outputs:

POKE6578,PEEK (56578) OR3.

Then to change banks enter:

POKE56576, (PEEK (56576) AND 252) OR (3-B) where B is the desired bank number in the range 0 to 3. The banks' start addresses are 0, 16384, 32768 and 49152. Let's utilise the ROM images of bank two, accessing the lower case character set, with the screen at 40960. First step as always is to find the offsets. These are 6144 for the lower case letters and 4096-32768=8192 for the screen memory. Dividing to produce "in range" answers gives 6144/2048=3 and 8192/1024=8. So simply get the correct bits which will be 011 (3 for the character data) and 1000 (8 for the screen memory). Now combine them with the screen memory fist and add a zero to give 10000110 – a value of 134 in decimal. Store this in location 53272 and hey presto! One small problem, though, and that is why I selected that example.

The screen memory is at 40960, the start of the BASIC Interpreter ROM. To quote from a little box on page 261 of the Programmer's Reference Guide – in any memory map containing ROM, a WRITE (a POKE) to a ROM location will store data in the RAM "under" the ROM. Writing to a ROM location stores data in the "hidden" RAM. For example, this allows a hi-resolution screen to be kept underneath a ROM and be changed with-

out having to bank the screen back into the processor address space. Of course a READ of a ROM location will return the contents of the ROM, not the "hidden RAM". (Unquote) What a mouthful! It basically means that where you may think there is only ROM, there is also memory that can be used by you that runs parallel to it. By POKEing to one of these locations you change the RAM but upon reading that same location you will read the contents of the ROM and not the RAM that you have just written to. If you still don't understand that then never mind – it's just another example of those blokes (and ladies) at Commodore making life difficult for the humble computer owner.

Therefore, you will be able to type text and graphics characters and they will appear fine because the computer is writing these to the "hidden" RAM (which, incidently is what the computer looks at for the screen information, as opposed to the ROM). Because it writes to the RAM and screen information that is displayed is taken from the RAM you are fine. Problems arise when you want to enter a direct command such as PRINT"HELLO" or POKE53280,2 or whatever. The interpreter will not read the information that you have typed from the correct screen memory area. Instead, it will read from the ROM and so a syntax error will be generated. The same happens if you move the cursor over what you have typed. The cursor flashes by storing a value of 160 (solid block) at the correct location and stores the character that was (under" the block. When it does this it reads what it thinks should be under the block (and therefore what it will replace when the cursor blinks) from the ROM instead of the RAM and so a whole load of jargon (computer term, wow!) appears in its place. Therefore avoid placing your screen memory where the computer thinks ROM will be, unless you switch it out by altering location one – that process can be explained by someone else as it has nothing to do with this article!

So now you will hopefully know how to change banks and select your character set and screen memory areas (if not then start reading again!). There is one final address that I have not even mentioned and so to avoid the old party trick of the User's Manual, I shall explain it in great detail. The location is 648 ($0288) and tells the computer where to look for and write its screen data to. It is the start of the screen memory divided by 256.

To prove this, you all know that the screen memory map usually starts at 1024, so read the contents of 648 with a simple PEEK command. You should get the result of four and key presto, 4x256=1024!

So with the screen memory area, character definition area and, most importantly, the bank number selected, you should finally change the value held in location 648 to the page at which your screen starts. This is the actual value and not just an offset. For instance, with the screen in bank three at location 49152, you would store the value 49152/256=192 in location 648.

However, if you are POKEing direct to the screen area you need not bother with the contents of 648. This particularly refers to multicolour pictures where it is likely that you will POKE the values direct to the screen memory area that you have selected. Usually whenever the computer displays something, whether it be one of the error messages or kernal messages such as "PRESS PLAY ON TAPE" or "LOADING", or whether it be from one of your PRINT statements, the characters are actually POKEd to the screen memory area by the computer according to the contents of location 648 and not the high nybble of 53272. This means that, taking my previous example of the screen at 32768 and the characters at 36864, you MUST enter POKE648,128 if you want normal messages to appear to you.

If you leave 648's contents at their default setting of four then whatever you type directly or print to the screen will be stored at the corresponding locations between 1024 and 2023. This can be illustrated using a monitor. type POKE648,192. The READY prompt that would usually follow such a command will not appear. Now type carefully POKE648,4 and a prompt will appear. use your monitor to view locations 49152 to 50151 and you will notice a number of $20 bytes which are the spaces and also the numbers that correspond to the READY prompt that was stored there instead of between 1024 and 2023. The effects of 648 can be demonstrated without the use of a monitor although it is more likely that you will crash the computer if you do not follow the precise instructions.

First of all clear the screen from power-up and type POKE648,5. The computer will then think that its screen memory starts at 5x256=1280. The READY prompt will then be stored at corresponding locations between 1280 and 2179. This will mean that the prompt will appear about six to nine lines down and to the right of the display. Immediately type POKE648,4 and press RETURN otherwise the computer is likely to crash. The latter will also occur if you store a value of less than 4 in location 648. For example POKE648,0 would crash the computer immediately as some of the $20 bytes that represent spaces and the screen codes for the READY prompt would be written to page zero which is of course the area that handles most of the computer important operations.

This then is all I plan to say about banking and memory selection. You have more than sufficient information to be going on with. Just remember the simple steps of dealing with 56576, 56578, 53272 and 648. So long as you can master these then you are well on the road to understanding memory area and bank number selections.

On the disk you will find a program filed as "BANKS AND MEMORY". This is a simple BASIC program that, upon you giving valid entries for the start of screen memory and character data, will produce a list on the screen of the necessary values that need to be POKEd where. It also tells you the addresses of the eight sprite pointers. Remember, with sprites it is only the pointers and range over which data can be viewed that are changed. The program will also allow you to enter hexadecimal numbers by prefixing the value with a dollar ($) sign. Should you choose bank zero or bank two for the screen memory, the computer will automatically display the location for the upper case ROM image. If you do not want to utilise it then simply overtype the value with the start of your character set or bit-map, again in either decimal or hex. I hope you will now find the whole topic a lot more comprehensible.

Graphics Factory

If you like to produce impressive looking intro screens or just like dabbling in graphics then this utility is right up your street

By Marco Westerweel

THE CONCEPTS

The idea behind a utility of this type is to maximise the C-64's graphics potential while minimising the amount of programming required to do so. To this end the program is designed rather like a word processor in that screens can be visualised as they are being created, thus freeing the mind for artistic inspiration rather than be burdened with the fatiguing effort of de-bugging string and poke algorithms till all hours of the morning.

"Graphics Factory" basically creates a screen file library which organises files into a central control menu from where a variety of file handling tasks are performed. Once saved to disk; such screens can then be copied to other disks and used in programs of the user's own design for virtually any purpose: i.e. intros, menus, instruction text, game and utility themes, score boards, etc.

There are several advantages to using screens created with "Graphics Factory". For one thing they download in mere seconds (once "CRUNCH"-ed from within the menu). Also they can easily be updated to reflect whatever happens while the program is running. In addition, using screen files tends to streamline one's programming style by reducing the number of PRINT and POKE statements, and encourages modularising the program into subroutines built around the theme logic of specific screens.

USING GRAPHICS FACTORY

Type: "GRAPHICS FACTORY"S and RUN, to get started. An intro screen will appear while the program initialises, after which you will find yourself in the top right corner of the operating menu from where all file manipulations are executed. The logic flow of the menus divided into two categories: NEW and OLD. NEW allows the user to create files and save them to disk, while OLD lists all existing files and provides a series of options for processing them: COPY, MODIFY, CRUNCH, REPLACE, SCRATCH and VIEW.

The following pages describe in detail how to use the operating menu commands:

NEW/OLD: When you first enter the menu, NEW is highlighted to indicate your starting point. If you want to start by creating a new file then hit the RETURN key, otherwise press CRSR-left which then highlights the OLD command and press RETURN to execute it. You get plenty of opportunity to change your mind and move around between choices since commands are not executed until they are RETURN-ed.

Selecting NEW will cause a question mark to continue flashing in the file name entrybox until a name is typed in and RETURN-ed. Any typos that may occur while entering the name can be deleted with the DEL key. After the name is entered a CANCEL/ACCEPT option is offered in case you want to start over. Here too the CRSR keys are used to highlight your choice and RETURN executes it.

If you select OLD at the start, then the menu proceeds to the directory listing box where the ADVANCE command is highlighted and the first six files listed. The directory contains 54 dummy titles indicated with an asterisk (*), these are changed to names supplied by the user whenever a screen file is saved to disk. RETURN-ing the ADVANCE command will list the next six files, while CRSR-left and RETURN executes the REVERSE command which lists the previous six files.

To select a file name for processing, CRSR_down from the ADVANCE/REVERSE box and the file names will highlight one by one each time CRSR_down is pressed. When you are at the file you want, RETURN it and you will automatically enter the options section. Options are also selected by CRSR-ing to them and hitting RETURN.

MODIFY: Loads an existing file and allows the user to edit and save it to disk under another name without erasing the original version. This is quite handy for standard background screens on which only a few details need to be changed to create a new application. Three such editable files (analogous to picture frames) are included on disk: "Shadow Box", "Wire

Frame" & "Fancy Wire Frame".

REPLACE: Also allows editing and saving existing files, but erases the original file and substitutes the newly edited version of it under the same name as the original.

CRUNCH: Converts editable files into non-editable files which occupy a lot less memory and download much faster. Such files are prefixed with "CR/" in the directory, and it is in this format that a file must be prior to being copied to another disk for use in your own programs. The original editable version remains intact for later use. This process takes almost three minutes because string ($) processing in BASIC accumulates a lot of garbage in RAM which must be cleared when there is no more room for it.

COPY: Allows the user to write files to other disks.

SCRATCH: Deletes old files. This rearranges the directory because the scratched file name is replaced with the last file name, and the last file is replaced with an asterisk.

VIEW: Enables the user to see a file without processing it. This is convenient for just browsing and avoids the possibility of accidentally deleting or changing a file that was not intended for that purpose.

CANCEL/ACCEPT: All menu operations pass through this box for final approval. CANCEL is highlighted first, thus if you want to start over then just press RETURN. CRSR-right highlights the ACCEPT option and RETURN will execute it.

EDITING SCREEN FILES:

Whether you are MODIFY-ing, REPLACE-ing, or creating a NEW file, the editing process is the same. A flashing cursor indicates your screen position, and the CRSR-keys move it around for formatting and editing. Graphic data is entered via the keyboard by making various combinations of the RVS ON, RVS OFF and 16 colour keys with the alphanumeric and graphics keys. A few characters such as commas, quotes, colons, RETURN, CLR/HOME, and INST/DEL are ignored by the editor because they create problems with file processing and disk storage. These characters can however still be used in programs with screen files.

There are four options listed at the bottom of the screen when in editing mode:

(F2) SAVE: Writes screenfile to disk in editable format.

(F4) GAR/COL: Forces garbage collection (clearing RAM) at your convenience. This takes one to two minutes, so it is nice to have this option for whenever you want a small break anyway. If you don't press F4 occasionally (once or maybe twice per hour for heavily detailed screens), then the program eventually freezes up for a minute or two so it can handle the garbage collection itself.

(F7) PLOT: Restores utility back to standard entry mode.

LOADING SCREENS INTO YOUR PROGRAMS

After you create screen you like, CRUNCH and COPY it from the work disk to the disk with your program on it. A subroutine named "GF/Screen Loader" is included on disk for loading and printing screen files in your programs. This nifty little BASIC routine can either be merged to your program if you have a merge utility of your own, or you can simply load it, then build up your program around it, and save it under a new name. Either way you never need to bother typing it in.

To read, load and print the screen file you must first specify the file name in FL$ as in the following example:

```
100 FL$="FILE NAME" GOSUB10000
110 GETGT$:IFGT$=""THEN110
```

Line 100 calls the subroutine and line 110 waits for any key to be pressed before continuing. If you need to reprint the screen later on then you can skip the read and load section and call the printing section of the subroutine with GOSUB11000.

GETTING IT ALL IN

Before using the utility, copy all the relevant files to a blank work disk with the CDU file copier. The entire "Graphics Factory" package consists of:

GRAPHICS FACTORY (BASIC):	Main program
CR/GF INTRO (SEQ):	Intro screen
CR/GF MENU (SQ):	Options menu screen
GF/DIR (SEQ):	File name directory
GF/SCREEN LOADER (BASIC):	Loads files into programs
SHADOW BOX (SEQ):	Sample screen file
WIRE FRAME (SEQ):	Sample screen file
FANCY WIRE FRAME (SEQ):	Sample screen file
CR/DOODLE (SEQ):	Sample screen file

Did you ever try to create 3D multicolour graphics for your BASIC programs with POKE statements? Chances are that if you did, the results were far from optimal and it took much longer than it was worth to do it. The demo slide show on disk is a preview of the kind of graphics power you can expect to have at your fingertips next month when the GRAPHICS FACTORY utility comes out.

GRAPHICS FACTORY is capable of creating eye dazzling illusions in medium resolution by imposing one plane above another. The top plane acts as a canvas, it consists of 23 lines of 38 inversely printed purple spaces. The bottom plane is the standard background at power up, changed in this case to light grey.

This two plane gimmick enables the BASIC programmer to achieve a considerable amount of leverage over standard POKE graphics in BASIC. For example: try to POKE a white playing card with a red heart and black rank against a blue background. It won't work because the background of the suit and rank will not be the same colour as the card. With GRAPHICS FACTORY on the other hand, things like that and lots of other tricks are a cinch.

With a little imagination you can incorporate scores of colourful and aesthetically pleasing screens into your own programs. The demo will give you plenty of ideas. Enjoy!

Utility Pot Pourri

We have put together a mixture of some interesting yet helpful small utilities to help keep your interest in the C64 alive

By M. Carroll

Utility programs come in many weird and wonderful forms. Some are large complex extended basics, whilst others are very short one liners. Most of us have at some stage in our programming careers used a vast variety of such utilities. The problem is normally remembering on which disks in our libraries certain utilities are. To alleviate this problem, I have put together 15 of these simple utilities which you can keep on a single disk, thus making access to them easy. I have chosen a fairly mixed bag of stuff and I hope that something takes your fancy. Most of the routines are located at 49152 ($C000) onwards, the usual place that programmers pick for their machine code routines.

These routines will not be shown on the CDU menu program, in order to use them you will have to load them directly yourselves.

Sideways Dump

This program basically allows your printer to print out text sideways, a rare facility, therefore giving you an unlimited paper width.

This program can be called from BASIC or machine code and can print text from anywhere in memory, as long as the text is in 4164 RAM, not I/O RAM. Only experienced users need worry about this. The program itself can also be easily relocated – including workspace it resides more $C000 to £C178. It should be compatible with most commodore printers; more about this later.

How it all works:-

Disassemble locations $C000 to $C15D. Throughout, the locations outside the code that are used are:-

$02 This location saves the mode number to be used.

$14-$15 Are used for converting FP to hex numbers.

$22-$23 Workspace for address calculations.

$8B-$8C General workspace.

$9E-$9F Used on machine code call to transfer parameters.

$FB-$FC Preserve the address of the block to be printed.

$FD-$FE Preserve the X and Y co-ordinates to be reached.

$02A7 Is used to preserve the contents of location 1.

And locations inside the workspace are:-

$C15E Current X co-ordinate reached.

$C15F Current Y co-ordinate reached.

£C160-$C178 General workspace and the ASCII to screen code conversion table.

The machine code itself runs as follows:-

$C000-$C00B Check to see if it being called from BASIC or machine code.

$C00C-$C02E Get parameters from BASIC.

$C02F-$C042 Get parameters from machine code.

$C043-$C04A Initialise one or two workspace locations.

$C04B-$CO6D Printer initialisation section.

$C06E-$C117 Print character side ways – this involves quite a lot.

$C118-$C121 Point to next horizontal row.

$C122-$C13D Point to next vertical column, and do a sort of carriage return on the printer.

$C13E-$C15A Reset printer and exit program.

$C15B-$C15D Jump back into main loop at $C06E.

$C15E-$C178 Workspace and ASCII to screen code conversion table.

Instructions:-

The program lies dormant in memory until called. To call it from basic use SYS49152,x,x,y,m

To call it from machine code, use:-

LDX #Lo-byte of parameter block.
LDY #Hi-byte of parameter block.
SEI
JSR $C000
CLI

The parameter block is formatted as follows:-

Byte 0 – s
Byte 1 – x
Byte 2 – y
Byte 3 – m

This machine code technique is used on the BBC micro OSWORD routines.

The parameters are as follows:-

s – The start address of the block to be printed sideways. For the normal screen this is 1024.

x – The number of columns the block has. The screen has 40.

y – The number of rows the block has. The screen has 25.

m – The mode you want the block printing in. These are:-

Mode number	ASCII/Screen codes	Upper/Lower case
0	Screen	Upper
1	ASCII	Upper
128	Screen	Lower
129	ASCII	Lower

So, for example, if you wanted an upper case screen dump you would use:-

SYS49152, 1024, 40, 25, 0 [Return]

And for a lower case screen dump you would use:-

SYS49152, 1024, 40, 25, 128 [Return]

To load the program either use:-

LOAD "SIDEWAYS DUMP" , 8, 1 [Return]
NEW [Return]
or, to avoid corrupting the basic program in memory, use:-
OPEN1, 8, 1, "SIDEWAYS DUMP, P, R": POKE780, 0: SYS65493: CLOSE1 [Return]
Once loaded, the program can be used anytime.

To read the disc directory you have to first save the program in memory, load the directory, check it, then load the program that was in the computer originally. This is immensely inconvenient, especially when you are working on a large program and you forget the filename you normally save it under. All these problems are solved with this small handy utility. This also gets past some disc protection, such as invisible directories, and link-bytes which loop back to Track 18 Sector 1.

How it all works:-

Dissassemble locations $C000 to $C05D. Throughout, the locations outside the code that are used are:-
$C05E Temporary storage for the accumulator.
$C05F The filename of the directory – the ASCII code for a '$'.
The code runs as follows:-

$C000-$C01C	Open an input file to the disc drive – the directory.
$C01D-$CO2D	Do a CTRL + 9 for the disc title.
$C022-$CO2F	Search for the first quote and I/O errors.
$C030-$CO49	Print filename or disc title, checking for other quote, and disc errors.
$C04A-$CO55	Print a carriage return and, check ing for errors, go back to $C022 for next filename.
$C056-$C05D	This closes the channel to the disc-drive and ter minates the pro

gram.
Instructions:-
Type:-
SYS49152 [Return]
and the directory will be displayed. Use RUN/STOP to halt the directory listing. Basic will remain untouched.

To load the program either use:-
LOAD "DIR", 8, 1 [Return]
NEW [Return]
or, to void corrupting the basic program in memory, use:-
OPEN1, 8, 1, "DIR,P,R": POKE780,0: SYS65493: CLOSE1 [Return]
Once loaded, the program can be used anytime.

This utility can be called anytime by pressing CTRL ≠ RETURN, after which you will be able to zoom through the whole memory altering any location you want (except for the ROM) then return to whatever the computer was doing before with, usually, no apparent change. You can call this utility whilst LISTing, LOADing, or most other things. To activate it, load it with:-

LOAD"MEMORY EDITOR", 8,1
NEW
SYS49152
Instructions:-

Load the program as shown above. From then on, unless the program is corrupted, you can use:-
SYS49152 [Return]
to ensure that it is activated, and CTRL + RETURN anytime to call the memory editor. When in the memory editor, use:-
Cursor up – Move low down in memory.
Cursor down – Move higher up in memory.
Cursor left – Move cursor left.
Cursor right – Move cursor right.
f1 – Zoom lower down in memory.
f7 – Zoom higher up in memory.
0 to 9 – Enter hex digits.
A to F – Enter hex digits.
RETURN – Exit memory editor.
Certain memory location, such as $02 are used by the memory editor. $FB and $FC are used to hold where the memory editor is editing in memory. Alter $FC for a rapid jump.

Autoverifier

This is quite useful when developing another program. Normally, if you wanted to SAVE a program, you may use '@:' to save it over the old one. This is risky. The alternative is to scratch it first. This is inconvenient, as is verifying the program afterwards if you are worried about it not saving properly. It is also annoying putting ,8 after each disk command. The need for all of this is eradicated with this program. It uses as a default on initialisation Unit 8 Drive 0. To change this, use:-
SYS52941, Unit, Drive
To put the program in operation, use:-
SYS52976
To make the Datasette (TM) the default device again, use:-
SYS65418

Whilst the program is in operation, adding a device number to the filename has no effect whatsoever – the device number and drive are now what was previously specified. Instead of using:-
SAVE "@0: Filename" ,8
you must now use:-
SAVE"Filename
at which point the computer will scratch the old file (if there was one), save the new file, then verify it. It is not uncommon for the computer not to say 'OK' after verifying; it has only failed if a LOAD ERROR or VERIFY ERROR is reported.

The LOAD command now uses the format as the SAVE command, and also has an automatic verify built in. there is only one problem – loading the directory. However, the DIR program, also written by me, should have been published by now, and it still works with this other program in operation.

Faster 64

This simple program allows the C64 mode of the C128 to take advantage of the 2MHz operating mode of the 8502 processor.

SYS49152,0 This turns the program off, equivalent to the SLOW command. Interrupts are returned to the KERNAL's control.
SYS49152,1 This, using raster interrupts, puts the processor in 2MHz mode in the border, 1MHz mode in the screen. This stops the display being upset (except in the case of 'sprite in the border' programs) and speeds the computer up to about one and a half times the normal operating speed. Interrupts are handled by the program.
SYS49152,2 This is equivalent to the FAST command. The screen is blanked, and the processor is forced into 2MHz mode. Using POKE 53265,27 will

'unblank' the screen, but I'm not sure if the VIC-II chip can take it. Interrupts are returned to the KERNAL's control.

This is totally compatible with the normal C64. The effects are as described above, but there is no change in the computer's operating speed. To use this from machine language, instead of using:-

SYS49152,n

from BASIC, use:-

LDA #n
JSR $C0A8

instead. If at the end of a routine, a JMP to the code is perfectly acceptable.

53 Column Display

This program adds an extra 13 columns to your C64's display without disturbing memory below $C000, except for some zero-page locations, and also provides some useful routines for manipulating this new screen.

Instructions:-

To load the memory block use the commands:-

OPEN1, 8, 1, "53 COLUMN DRIVER,P,R"
POKE780,0
SYS65493
CLOSE1

After that several commands are available, even after resetting the computer with 'SYS64738". These are:-

SYS49152,0 This command clears the 53 column screen, but leaves the 53 column cursor position unaltered.

SYS49152,1 This command transfers the 53 column cursor position into the keyboard buffer. To read it into the variables X5 and Y5 use:-

SYS49152,1: GETX$,Y$
X%=ASC(X+CHR$(0))
Y%=ASC(Y$+CHR$(0))

SYS49152,2,x,y This command changes the 53 column cursor position. x may be from 0 to 52, and y from 0 to 24. To move the cursor to the top left-hand corner of the screen use:-

SYS49152,2,0,0

SYS49152,3,n,c,i This command prints a character at the current cursor position. The parameters are:-

n – the screen code of the character to be printed, from 0 to 127.

c – This can only be 0 or 1, denoting the case to be used – 0 for upper case, 1 for lower case.

i – This, also, can only be 0 or 1. If it is 1 then the character is inverted before being displayed, as normal characters are after pressing CTRL + 9.

For example, to print a lower case letter Z in normal type, use:-

SYS49152,3,26,1,0

SYS49152,4,m This command changes the screen mode. The options are:-

SYS49152,4,0 for 40 column.
SYS49152,4,1 for 53 column.

SYS49152,5 This command scrolls the 53 column screen upwards one character. This is not automatic if the bottom right-hand corner of the screen is printed in, and must be done manually.

How it all works:-

When this program is used the memory is as follows:-

$C000-$C310 Main code – detailed more later.
$C311 – $C710 Upper case character definitions.
$C711–$CB10 Lower case character definitions.
$CC00-$CFE7 Colourmap for the 53 column screen.
$E000-$FF3F The 53 column (hires) screen.

The code is organised as follows:-

$C000-$C008	Pick up first Basic parameter.
$C009-$C025	Jump to the routine request by this parameter.
$C026-$C031	The addresses of these routines.
$C032-$C040	The start of the 'clear screen' routine.
$C041-$C04F	Main loop and routine termination.
$C050-$C060	The routine for placing the current X and Y co-ordinates into the keyboard buffer.
$C061-$C087	The routine for chain the cursor position.
$C088-$CoB8	This piece of code picks up the parameters for the routine to print characters, and starts calculations.
$C0B9-$COE4	This completes the first set of calculations.
$C0E5-$COFO	This transfers the bit pattern of the character definition into the workspace area.
$COF1-$C101	If necessary, this inverts the character.
$C102-$C158	This completes the second set of calculations.
$C159-$C160	This holds the addresses of the routines for completing the character-writing process. Four of these are necessary.
$C161-$C179	The first routine.
$C17A-$C1D6	The second routine.
$C1D7-$C222	The fourth and last routine.
$C33E-$C25E	This completes the character-writing process, sorts out the new X and Y co-ordinates, and returns to Basic.
$C25F-$C279	This starts the changes by the 'change screen mode' routine and decides which screen is requested.
$27A-$C28C	The routine for switching to 40 column.
$C28D-$C2C6	The routine for switching to 53 column, which has to sort out the colourmap.
$C2C7-$C2D4	This starts up the upwards scrolling routine.
$C2D5-$C2EF	This shifts all the bytes backwards and initialises the next part of the routine.
$C2F0-$C2FE	This completes the routine by clearing the bottom line and returning to Basic.
$C2FF	The X co-ordinate. It can be read directly from this location.

$C300	The Y co-ordinate. It can be read directly from this location.
$C301-$C310	This is workspace for manipulating the character bit pattern.
$C311	onwards The character definitions etc.

To check all the definitions use the following program:-

```
10 SYS49152, 0: SYS49152,0,0
20 SYS49152,4,1: FORA = 0TO1: FOR B= 0 TO1
30 FORC = 0TO127: SYS49152, C, B, A
40 NEXT: NEXT: NEXT: POKE198, 0: POKE 198,1
50 POKE198,0: SYS49152, 4, 0
```

After running this program, press any key to return to the normal screen. To change a definition you need to know its base address. To calculate this, use the formula:-

Address=$ C311+ 8* (Screen code)

Add on $800 for the low case equivalent. The eight bytes starting from this position hold the character definition. However, these are in fact on a 6 x 8 matrix, so the upper two bits are ignored. Incidentally, the last two pixels on each line of the hires screen always remain blank as 53 columns does not fit exactly onto the screen. The only other thing worth mentioning is the sequence which occurs regularly in the program, that of three JSRs to the Basic Interpreter. Their meanings are:-

JSR $AEFD Check for a comma in the Basic line being processed.
JSR $AD8A Pick up a number from the Basic line and place it in FAC #1.
JSR $B1AA Transfer the contents of FAC #1 to the Accumulator and Y-Register, the least significant byte in the Y-Register.

Register Report

This program maintains a constant display of the major 6510A/8502 registers. This can be useful when trying to find out where machine-code is getting stuck in an infinite loop or something. The display is like the C128's brief report if you turn it on with the RUN/STOP key held down, except that the different flags of the status register are displayed separately.
Usit it:-
To start the program running, type:-
LOAD "REGISTER REPORT", 8,1 [Return]
SYS49152 [Return]
To pause it, allowing it to be removed from the screen, use:-
SYS65418 [Return]
Typing SYS49152 will restart it again.

Explanation of display

The display should be as follows:-
NV-BDIZC AC XR YR SP PC
xxxxxxxx xx xx xx xx xxxx

These are as follows:-

N	This shows the negative flag's status.
V	This shows the status of the overflow flag.
–	This should be a 1. It is unused.
B	This is a 1 if a BREAK occurs.
D	This is the decimal flag. It's a 1 if BCD is being used.
I	This is the IRQ flag – this should be a 0.
Z	This reflects the zero flag's status.
C	This shows the carry flag's status.
AC	This shows the contents of the accumulator.
XR	This shows the contents of the X index register.
YR	This shows the contents of the Y index register.
PC	This shows the program counter's value – the address at which machine-code is cur rently being executed.

How it all works:-
The program is organised as follows:-

$C000-$C00C	This changes the interrupt vector to start the display up.
$C00D-$C031	This cleans up the screen display by redisplaying the constant charac ters of the display.
$C032-£C052	This picks up all the registers whose statuses are to be dis played.
$C053-$C061	This displays the status register's flags.
$C062-$C088	This displays the hex numbers re quired, using the routine at $COD5
$CO89-$COA43	This puts every thing back on the stack and jumps to the normal IRQ vector.
$C0A4-$COA5	The temporary storage for the program counter.
$COA6	Storage for the accumulator.
$COA7	Storage for the X index register.
$COA8	Storage for the Y index register.
$C0A9	Storage for the stack pointer.
$COAA	Storage for the status register.
$COAB	Workspace for the status register.
$COAC-$COC4	The top line of the display, in screen codes.
$COC5-$COD4	The hex digits, in screen codes.
$COD5–$COEE	The routine for 'printing' the hex numbers.
$COEF	The first free byte after the routine.

The whole display runs on an interrupt.

Key Definer

There are many utilities around to redefine function keys – some boast 12 redefinable keys. This program redefines up to 56 keys. To activate it, use SYS49152. To clear the definitions use SYS49788 – use this when first loading the program. From then on, use (in direct mode):-

*KEYn, "Text"

as you would on a BBC micro. Use the back arrow as a carriage return, as when the key is used, the string is actually typed. The key numbers are as follows:-

0CTRL + RETURN Note:-
1-26CTRL + LettersDefinitions 0-31 use

memory
27CTRL + from $A000 to $BFFF.
28CTRL + [Definitions 32-55 use memory
29CTRL +] from $E000 to $F7FF.
30 CTRL + =
31CTRL + £
32-35f-Keys
36-39SHIFT + f-Keys
40-43Commodore key + f-keys
44-47CTRL + f-Keys
48-51CTRL + SHIFT + f-Keys
52-55 CTRL + Commodore kye + f-Keys
To load the machine-code initially, use:-
LOAD "KEY DEFINER", 8,1
NEW

Note Calculator

Whenever you want to put a soundtrack in a game, or jut play a tune on the C64 there are lots of tedious numbers to work out and poke, based on Appendix M of the User Guide, or Appendix E of the Reference Guide. This involves many boring DATA statements, each number carefully picked out from the manual's note tables. To make it worse, as you start to develop bleary eyes and aching limbs you make some typing mistakes or a few errors in looking a the tables – errors which you have to look for afterwards amongst a mass of numbers. Until now. This program works out all these numbers for you – you just say which notes you want.
To load it use:-
LOAD "NOTE CALCULATOR" , 8, 1 [Return]
NEW [Return]
or, alternatively,
OPEN1, 8, 1, "NOTE CALCULATOR": POKE780,0: SYS65493: CLOSE1 [Return]
From then on, to output a frequency on a voice, set the other POKEs, like the ADSR and volume, to whatever you want, then use:-
SYS49152,n,a,0,v
The parameters are as follows:-
n – This is the number of the note you want. these are:-
C – 0
D – 1
E – 2
F – 3
G – 4
A – 5
B – 5
If working out this number for the note is too much for you (!) then the following formulae can be used:-
N$ = The letter (or note) for the number.
n = The number for the letter (or note).
N$=CHR$ (67 = n 7* (n>4))
n= ASC (N$) – 67 - 7* (ASC(N$)<67)
a– The accidental of the note. Notes are either sharp, flat, or natural. The numbers are:-
:a:Accidental:Sign:
:0:Natural::
:1:Sharp:#:
:2:Flat:b:
0 – This is simply the octave of the note required, from 0 to 7. Middle C is the C of octave 4.
v - This is the voice that the frequence of the note is to be given to. This can be from 0 to 2.
For example, to play middle C on voice 0, use:-
SYS49152,0,0,4,0 [Return]
For a basic test of the program, set up voice 0 with:-
FORA = 0TO24: POKE 54272 + A, 0: NEXT [Return]
POKE54296, 15: POKE54277,17: POKE 54278, 255: POKE 54276,17 [Return]
Now any frequency given to voice 0 can be played, such as middle C previously demonstrated.
How it works:-
The routine at $C07E picks up a BASIC parameter. The machine code calculates, from the note number and accidental, the number attributed to the octave 0 notes on page 384 of the Reference Guide, from 0 to 11. The octave is then multiplied by 12 in the floating-point accumulators, and added to the last number calculated. The base address of the table of note values is then added on, resulting in the final number pointing to the double byte address of the frequency required.

File Protector

Sometimes is it desirable to stop people looking at your work. Many techniques exist but most involve you doing something to your program before saving, such as calling some machine code that mucks around with memory at the beginning, or scrambles the BASIC. This is a little annoying if you want to edit the program later, as the protection must be stripped first. This program does strange things to the file as it is saved or loaded to prevent other people loading it. If you prevent people loading the file in the first place, you do not have to bother protecting the actual coding. Only experienced disc users will see what has been done to stop any loading of the file.
To load the program, use:-
LOAD "FILE PROTECTOR",8,1

Instructions:-
To activate the program, use SYS49152. To deactivate it, and return to normal I/O operations, use RUN/STOP and RESTORE, or SYS65418. When the program is activated, the following commands are active:-

LOAD "Filename",dn
SAVE "Filename",dn
VERIFY "Filename",dn
SYS49370 "Filename",dn

where the filename is as you would normally use it, and dn is the device number. If no device number is given, or the device number is not a disc-drive, then it is forced to device 8. The last command is for scratching a protected file. No comma exists between the 0 and the quotes.

A quick test:-
Load the program then enter the following:-
: Command: Result:
: 10 PRINT "HELLO": Nothing:
: SYS49152: Activates program:
: SAVE "TEST": Saves file (protected):
: SYS65418: Deactivates program:
: LOAD "TEST",8:?FILE NOT FOUND ERROR:
: SYS49152: Reactivates program:
: LOAD "TEST": Program loads successfully:
: VERIFY "TEST": Verifies correctly:
: SYS49370 "TEST": Scratches testfile:
:SYS65418: Deactivates program:

Revealing how this program works defeats the object of using it, so no explanation will be given in this article.

Program Scrambler

To load it, use:-

OPEN1, 8, 1, "0: PROGRAM SCRAMBLE, P, R" : POKE780,0:SYS65493: CLOSE 1
Then to activate it, use:-
SYS679
This program will enable you to scramble and de scramble the BASIC program currently in memory. It may be used in conjunction with the File Protector. SYS679 scrambles a BASIC program, or descrambles a scrambled one. It works out what to do for itself, therefore if you type SYS679 twice then the BASIC program would be scrambled then de scrambled, effectively leaving it unaltered. Revealing the technique used to scramble the BASIC program would defeat the object of using the scrambler – anybody would be able to descramble it again.

Drive Disable

A long while ago, in Your Commodore, there was an article on protecting programs from cartridges. The real problem is that once the freeze button is pressed, the computer is under the cartridge's control, so it is difficult to execute any of your own code at this point. It is, however, possible to execute code in the 1541c, and most other Commodore disc-drives, and to alter their memory without the cartridge noticing the change when the game or program is copied. This is, therefore, restricted to disc-drive users. One obvious way to do this is to use the M-W command to place a byte somewhere in the drive's RAM when the program is first loaded, than to keep checking for it. This, however, slows down the program. Saying how this program works would only defeat the object of using it, so I will only show how to use it. You put the program at the beginning of the main game, before the game runs. The code can then be disposed of, as it has already done its work. There are three calls to the code from basic that are possible:-
SYS53149,0 This renders the disc-drive unable to use the disc currently in the drive. This defeats things like Freeze Machine, although I have not yet acquired Action Replay MkV professional. It is possible to undo this command.
SYS53149,1 This deactivates the first call, in case there are files to be loaded. You can reactivate the protection with the first call.
SYS53149,2 This knocks out the whole serial bus. It is impossible to undo this command without turning the disc-drive off then on again – not the sort of thing you do with the computer on.

After using the last call you are going to, you can corrupt the actual machine code too, as the effects are in the disc-drive's RAM and VIA, not the computer's RAM or CIA.
To load the machine-code initially, use:-
LOAD "DRIVE DISABLE",8,1
NEW

File recovery

This program is for recovering lost files of damaged disks and putting them onto fresh ones. It works by copying the disk onto a blank one then scanning the blank one, reconstructing a new directory from what the program can find left on the disk. This is so that you can still use any other methods you have of recovering data off damaged disks – this program leaves the original disk unaltered.

To use it, simply select from the CDU menu, then enter the start addresses of any files you want to recover. For the uninitiated, some examples are:

0801 – C64 BASIC programs
2000 – HIRes screens
c000 – Machine code programs
1c01 – C128 BASIC programs
8000 – EPROM images

For example, if there were HiRes screens and BASIC programs on a disk that was damaged, you would type:-

0801 1c01 2000 [Return]

The program will then format a fresh disk, and do a four-pass copy onto it. If your damaged disk contains corrupted data then this process will not take long, but a physically damaged disk can take quite a while. After this, the program will construct a totally new directory track, then load in the directory for you. The filenames are of the following format on the directory:

aaa-bbb at ccccc

where aaa is the file number, bbb is the program's attempt at separating the files into machine-code (m/c) and BASIC (bas), and ccccc is the start address in decimal, so for example:

004-m/c at 49152

would be file 4 – machine code starting at 49152. Sometimes, machine-code programs are accidentally labelled 'bas'.

Now what you must do is to load each file in turn (some will not load properly – ignore them) then decide what exactly it is then save it on ANOTHER disk. When you have finished, reformat the fresh disk and use it for something else. For the technology minded, the program does a 4-pass copy locating the chunk of disk from $5A00 to $FFFF (skipping track 18) then searches the beginning of each block for the start address and any BASIC link-bytes. The arrays are:-
d(t,s) = -1 if a track are sector are valid.
s(b) = the directory sector currently being constructed.
a(n) = the start addresses to be looked out for.

Write-Protect

When testing a program that involves disc operations it is quite possible for a bug to creep in whereby the disc is written to accidentally, thus overwriting potentially important data. The golden rule is, of course, to make a backup before testing ANYTHING but it can be quite easy to put the wrong disc in – one that you did not make a backup of. This program prevents the disc drive from writing to a disc, and to undo this you repeat lines 10 and 80.

Getting it in:-

There's nothing to it. Just incorporate the listing in part of the program under test. The reversed 'A' and 'B' are obtained with CTRL + A and CTRL + B respectively.

How it works:-

This program merely fools the drive's second 6522 VIA into thinking that the write-protect sensor is returning a 1 by forcing it to output a 1 – the status register of the chip does not discriminate between inputs and outputs. full data of the chip's registers can be obtained from a good BBC Micro manual, as it uses the same chip. The C64 uses the upgrade, the 6526 CIA, so some of the registers will have the same function enabling you to use the Reference Guide as a means of finding the 6522's registers.

Commodore DISK USER

Lineage: 53p per word. (+ VAT)
Semi display: £11.50 plus VAT per single column centimetre minimum 2cm. Ring for information on series bookings/discounts.
All advertisements in this section must be prepaid.
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request).
Make cheques payable to ASP Ltd.

Send your requirements to:
CLASSIFIED DEPARTMENT
ASP LTD. ARGUS HOUSE,
BOUNDARY WAY, HEMEL HEMPSTEAD HP2 7ST.

0442 66551

SOFTWARE

SILVER WING SOFTWARE
'THE DEDICATED C64 P.D. LIBRARY
We have the best & latest quality demos, utilities & games available.
ONLY £2.00 on Tape or Disk
For the best benefits & power from your 64, you can not do without Public Domain Software...
For our lists send an SAE to:
SILVER WING SOFTWARE
185 CALLOWBROOK LANE,
RUBERY,
BIRMINGHAM
B45 9TG.

PUBLIC DOMAIN SOFTWARE
for the C64/128.
We have 150 disks full of all types of Programs
All disks are £2.95 each
NOW FOR THE +4
from £3.75 per disk.
Send SAE or phone for free catalogue (state model)
Kingsway Computer Services
72 Glencoe Road, Sheffield
S2 2SR Tel: (0742) 750623

REPAIRS

M.C.E. SERVICES
33, Albert Street, Mansfield, Notts.
NG18 1EA Tel: 0623 653512
FIXED PRICE COMMODORE REPAIRS

C64	£24.95
C16	£19.95
PLUS4	£24.95
AMIGA A500	£39.95
1541	£29.95
1571	£31.95
C2N etc	£12.95

All prices inc. VAT, Labour, Parts etc.
NO MORE TO PAY.
Please ring for details.

FULL TERMS AND CONDITIONS ARE AVAILABLE ON REQUEST CALL 0442 66551

DISKS

WESTONING LTD

3½ DISKS		5¼ DISKS		AMIGA	
DS/DD	46p	DS/DD	23p	BATMAN PACK	£355
DS/HD	£1.15	DS/HD	57p	FLIGHT FANTASY PK	£355
				CLASS OF 90's	£510
				512K RAM	£56
				WITH CLOCK	£62

PRICE EACH INC VAT AND LABELS

LOCKABLE BOXES	
CAP	CAP
40 - £4	80 - £5.75

PRICE EACH INC VAT, SLEEVES AND LABELS

LOCKABLE BOXES	
CAP	CAP
50 - £5.25	100 - £6.50

NEC EXTERNAL
3½ DRIVE ... £60
1084D MON £239

PRICES INCLUDE VAT ***NO QUIBBLE GUARANTEE***
POST ADD £3 ***NEXT DAY ADD £7***

12 SANDERSON ROAD, WESTONING, BEDFORD MK45 5JY
Tele. (0836) 775060 or (0525) 718668 Fax (0525) 715789

SERVICES

COMMODORE SUPPLIES

C64 Power Unit	£21.99
C128 Power Unit	£39.99
C2N/C3N Datasette	£24.99
100 x 3½" Discs	£29.99
C64 Magnum Lightgun	£34.99
C64/128 Serial Cable	£5.99

Prices include VAT & P&P Chq/PO to:
OMNIDALE SUPPLIES (CD),
23 Curzon Street, Derby, DE1 2E6
Tel 0332 291219

CALL 0442 66551 TO ADVERTISE

Commodore DISK USER

Lineage: **53p** per word (+ VAT)
Semi Display: **£11.50** (+ VAT) per single column centimetre. Min size 2cm.
Series discounts available on request. All classified advertisements must be pre-paid.
There are no reimbursements for cancellations.
SEND TO CDU ADVERTISEMENT DEPARTMENT, ARGUS HOUSE, BOUNDARY WAY, HEMEL HEMPSTEAD, HP2 7ST

I enclose my Cheque/Postal Order for £ ..... for ..... insertions, made payable to Argus Specialist Publications.
(Delete as necessary)

PLEASE DEBIT MY ACCESS/BARCLAYCARD NO

☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ **EXP. DATE** .....

£ .....**FOR** .....**INSERTIONS**

Name .....**Postcode** .....

Daytime Tel No. ..... **Signature** ..... **Date** .....

☐ **FOR SALE** ☐ **SOFTWARE** ☐ **SPECIAL OFFERS** ☐ **REPAIRS** ☐ **HARDWARE** ☐ **DISKS**

...it's dynamite!

POWER CARTRIDGE

FOR YOUR COMMODORE 64/128

"...unbelievable value for money" ZZAPP! Dec 89

- POWER TOOLKIT
- POWER MONITOR
- TAPE & DISK TURBO
- PRINTERTOOL
- POWER RESET
- TOTAL BACKUP

..."Money well spent" YC/CDU Jan 90

42 Pg Manual - "Damned Good Handbook" CCI Jan 90

SO MUCH FOR SO LITTLE

AVAILABLE FROM ALL GOOD COMPUTER RETAILERS

TRIED AND TESTED - OVER 100,000 SOLD IN EUROPE

..."highly recommended for C64 users" CCI Jan 90

YOU WILL WONDER HOW YOU EVER MANAGED WITHOUT IT

POWER TOOLKIT

A powerful BASIC-Toolkit (Additional helpful commands) that considerably simplifies programming and debugging.

AUTO	HARDCAT	RENUMBER
AUDIO	HARDCOPY	REPEAT
COLOR	HEX$	SAFE
DEEK	INFO	TRACE
DELETE	KEY	UNNEW
DOKE	PAUSE	QUIT
DUMP	PLIST	MONITOR
FIND	ILOAD	BLOAD

RENUMBER : *Also modifies all the GOTO's GOSUB's etc. Allows part of a program to be renumbered or displaced.*
PSET : *Set up of printer type.*
HARDCAT : *Prints out Directory.*

The toolkit commands can be used in your programs.

DISK TOOL

Using POWER CARTRIDGE you can load up to 6 times faster from disk.
The Disk commands can be used in your own programs.

DLOAD	DVERIFY	DIR
DSAVE	MERGE	DEVICE
DISK		

MERGE : *Two BASIC programs can be merged into one.*
DISK : *With DISK you can send commands directly to your disk.*

TAPE TOOL

Using POWER CARTRIDGE you can work up to 10 times faster with your data recorder. The Tape commands can be used in your own programs.

LOAD	SAVE	VERIFY
MERGE	AUDIO	

POWERMON

A powerful machine language monitor that is readily available and leaves all of your Commodore memory available for programming.
Also works in BASIC-ROM, KERNAL and I/O areas.

A	ASSEMBLE	I INTERPRET	S SAVE
C	COMPARE	J JUMP	T TRANSFER
D	DIS-ASSEMBLE	L LOAD	V VERIFY
		M MEMORY	W WALK
F	FILL	P PRINT	X EXIT
G	GO	R REGISTER	$ DIRECTORY DOS Commands
H	HUNT		

PRINTERTOOL

The POWER CARTRIDGE contains a very effective Printer-Interface, that self detects if a printer is connected to the Serial Bus or User-Port.
It will print all Commodore characters on Epson and compatible printers.
The printer-interface has a variety of set-up possibilities. It can produce HARDCOPY of screens not only on Serial printers (MPS801, 802, 803 etc) but also on Centronic printers (EPSON, STAR, CITIZEN, PANASONIC, etc).
The HARDCOPY function automatically distinguishes between HIRES and LORES. Multi-colour graphics are converted into shades of grey. The PSET functions allow you to decide on Large/Small and Normal/Inverse printing.
The printer PSET functions are:

PSET 0 - *Self detection Serial/Centronics.*
PSET 1 - *EPSON mode only.*
PSET 2 - *SMITH-CORONA mode only.*
PSET 3 - *Turns the printing 90 degrees!!*
PSET 4 - *HARDCOPY setting for MPS802/1526.*

PSET B - *Bit-image mode.*
PSET C - *Setting Lower/Upper case and sending Control Codes.*
PSET T - *All characters are printed in an unmodified state.*
PSET U - *Runs a Serial printer and leaves the User-port available.*
PSET Sx - *Sets the Secondary address for HARDCOPY with Serial Bus.*
PSET L1 - *Adds a line-feed, CHR$ (10), after every line.*
PSET L0 - *Switches PSET L1 off.*

POWER RESET

On the back of the POWER CARTRIDGE there is a Reset Button. Pressing this button makes a SPECIAL MENU appear on the screen.
This function will work with many programmes.

CONTINUE - *Allows you to return to your program.*
BASIC - *Return to BASIC.*
RESET - *Normal RESET.*
TOTAL BACKUP - *Saves the contents of the memory onto a Disk. The program can be reloaded later with BLOAD followed by CONTINUE.*
DIS' RESET ALL - *RESET of any program.*
TOTAL BACKUP TAPE - *As BACKUP DISK but to TAPE.*
HARDCOPY - *At any moment, prints out a Hardcopy of the screen. Using CONTINUE afterwards you can return to the program.*
MONITOR - *Takes you into the Machine language Monitor.*

BDL

Bitcon Devices Ltd

88 BEWICK ROAD
GATESHEAD
TYNE AND WEAR
NE8 1RS
ENGLAND

Tel: 091 490 1975 and 490 1919 Fax 091 490 1918
To order: Access/Visa welcome - Cheques or P/O payable to BDL
Price: £16.99 incl. VAT.
UK orders add £1.20 post/pack total - £18.19 incl. VAT.
Europe orders add £2.50. Overseas add £3.50
Scandinavian Mail Order and Trade enquiries to: Bhiab Elektronik, Box 216, Norrtalje 76123, SWEDEN. Tel: + 46 176 18425 Fax: 176 18401
TRADE AND EXPORT ENQUIRIES WELCOME